CATEGORY:

# CLEARED

ADDRESS
CONTACT IF FOUND:

| FORM PTO-1390<br>REV. 5-93 | US DEPARTMENT OF COMMERCE<br>PATENT AND TRADEMARK OFFICE | JC06 Rec'd PCT/PTO 2 2 JAN 2001<br>ATTORNEYS DOCKET NUMBER<br>P00,1950 |
|---|---|---|
| **TRANSMITTAL LETTER TO THE UNITED STATES**<br>**DESIGNATED/ELECTED OFFICE (DO/EO/US)**<br>**CONCERNING A FILING UNDER 35 U.S.C. 371** | | U.S. APPLICATION NO. (if known, see 37 CFR 1.5)<br>**09/744225** |

| INTERNATIONAL APPLICATION NO.<br>**PCT/DE99/01999** | INTERNATIONAL FILING DATE<br>**01 JULY 1999** | PRIORITY DATE CLAIMED<br>**27 JULY 1998** |
|---|---|---|

TITLE OF INVENTION
   **METHOD, ARRANGEMENT AND SET OF A PLURALITY OF ARRANGEMENTS FOR ELIMINATING AT LEAST ONE INCONSISTENCY IN A DATABASE COLLECTION CONTAINING A DATABASE AND AT LEAST ONE COPY DATABASE OF THE DATABASE**

APPLICANT(S) FOR DO/EO/US                **MICHAEL BERGER**

Applicant herewith submits to the United States Designated/Elected Office (DO/EO/US) the following items and other information:

1. ☒   This is a **FIRST** submission of items concerning a filing under 35 U.S.C. 371.
2. ☐   This is a **SECOND** or **SUBSEQUENT** submission of items concerning a filing under 35 U.S.C. 371.
3. ☒   This express request to begin national examination procedures (35 U.S.C. 371(f)) at any time rather than delay.
4. ☒   A proper Demand for International Preliminary Examination was made by the 19th month from the earliest claimed priority
        date.

5. ☒   A copy of International Application as filed (35 U.S.C. 371(c)(2)) - drawings attached.
   a. ☒   is transmitted herewith (required only if not transmitted by the International Bureau).
   b. ☐   has been transmitted by the International Bureau.
   c. ☐   is not required, as the application was filed in the United States Receiving Office (RO/US)
6. ☒   A translation of the International Application into English (35 U.S.C. 371(c)(2) - drawings attached.

7. ☒   Amendments to the claims of the International Application under PCT Article 19 (35 U.S.C. §371(c)(3))
   a. ☐   are transmitted herewith (required only if not transmitted by the International Bureau).
   b. ☐   have been transmitted by the International Bureau.
   c. ☐   have not been made; however, the time limit for making such amendments has NOT expired.
   d. ☒   have not been made and will not be made.

8. ☐   A translation of the amendments to the claims under PCT Article 19 (35 U.S.C. 371(c)(3)).

9. ☒   An oath or declaration of the inventor(s) (35 U.S.C. 371(c)(4)).

10. ☒   A translation of the annexes to the International Preliminary Examination Report under PCT Article 36 (35 U.S.C.
        371(c)(5)).

Items 11. to 16. below concern other document(s) or information included:
11. ☒   An Information Disclosure Statement under 37 C.F.R. 1.97 and 1.98; **(PTO 1449, Prior Art, Search Report, 06 References).**

12. ☒   An assignment document for recording. A separate cover sheet in compliance with 37 C.F.R. 3.28 and 3.31 is included.
             **(SEE ATTACHED ENVELOPE)**

13. ☒   Amendment "A" Prior to Action and Appendix "A".
    ☐   A SECOND or SUBSEQUENT preliminary amendment.

14. ☒   A substitute specification and substitute specification mark-up.

15. ☒   A change of address letter attached to the Declaration.

16. ☒   Other items or information:
        a. ☒ Request for Approval of Drawing Additions

        b. ☒ Appointment of Associate Power of Attorney

        c. ☒ EXPRESS MAIL #EL655302713US dated January 22, 2001

| U.S. APPLICATION NO. (if known, see 37 C.F.R. 1.5)<br>09/744225 | INTERNATIONAL APPLICATION NO.<br>PCT/DE99/01999 | ATTORNEY'S DOCKET NUMBER<br>P00,1950 | |
|---|---|---|---|
| 17. ☒ The following fees are submitted: | | CALCULATIONS | PTO USE ONLY |

**BASIC NATIONAL FEE (37 C.F.R. 1.492(a)(1)-(5):**

Search Report has been prepared by the EPO or JPO ................ $860.00

International preliminary examination fee paid to USPTO (37 C.F.R. 1.482) .. $690.00

No international preliminary examination fee paid to USPTO (37 C.F.R. 1.482) but international search fee paid to USPTO (37 C.F.R. 1.445(a)(2) .......... $710.00

Neither international preliminary examination fee (37 C.F.R. 1.482) nor international search fee (37 C.F.R. 1.445(a)(2) paid to USPTO ................... $1000.00

International preliminary examination fee paid to USPTO (37 C.F.R. 1.482) and all claims satisfied provisions of PCT Article 33(2)-(4) ................ $ 100.00

| | CALCULATIONS | PTO USE ONLY |
|---|---|---|
| **ENTER APPROPRIATE BASIC FEE AMOUNT =** | $ 860.00 | |
| Surcharge of $130.00 for furnishing the oath or declaration later than ☐ 20 ☐ 30 months from the earliest claimed priority date (37 C.F.R. 1.492(e)). | $ | |

| Claims | Number Filed | | Number Extra | Rate | | |
|---|---|---|---|---|---|---|
| Total Claims | 23 | - 20 = | 3 | X $ 18.00 | $ 54 .00 | |
| Independent Claims | 03 | - 3 = | 0 | X $ 80.00 | $ | |
| Multiple Dependent Claims | | | | $270.00 + | $ | |
| | | **TOTAL OF ABOVE CALCULATIONS =** | | | $ 914.00 | |
| Reduction by ½ for filing by small entity, if applicable. Verified Small Entity statement must also be filed. (Note 37 C.F.R. 1.9, 1.27, 1.28) | | | | | $ | |
| | | | | **SUBTOTAL =** | $ 914.00 | |
| Processing fee of $130.00 for furnishing the English translation later than ☐ 20 ☐ 30 months from the earliest claimed priority date (37 CFR 1.492(f)). | | | | + | $ | |
| | | | | **TOTAL NATIONAL FEE =** | $ 914.00 | |
| Fee for recording the enclosed assignment (37 C.F.R. 1.21(h). The assignment must be accompanied by an appropriate cover sheet (37 C.F.R. 3.28, 3.31). $40.00 per property | | | | + | | |
| | | | | **TOTAL FEES ENCLOSED =** | $ 914.00 | |
| | | | | | Amount to be refunded | $ |
| | | | | | charged | $ |

a. ☒   A check in the amount of $ 914.00   to cover the above fees is enclosed.

b. ☐   Please charge my Deposit Account No. _____ in the amount of $ _____ to cover the above fees. A duplicate copy of this sheet is enclosed.

c. ☒   The Commissioner is hereby authorized to charge any additional fees which may be required, or credit any overpayment to Deposit Account No. 50-1519.  A duplicate copy of this sheet is enclosed.

NOTE: Where an appropriate time limit under 37 C.F.R. 1.494 or 1.495 has not been met, a petition to revive (37 C.F.R. 1.137(a) or (b)) must be filed and granted to restore the application to pending status.

**SEND ALL CORRESPONDENCE TO:**

SCHIFF HARDIN & WAITE
PATENT DEPARTMENT
6600  Sears Tower
233 South Wacker Drive
Chicago, Illinois  60606-6473

_Mark Bergner_
**SIGNATURE**

Mark Bergner
**NAME**

45,877
**Registration Number**

## CERTIFICATE OF MAILING BY EXPRESS MAIL

**Express Mail Mailing Label Number   EL655302713US**

**Date of Deposit:    January 22, 2001**

I hereby certify that this correspondence is being deposited with the United States Postal "Express Mail Post Office to Addressee" service under 37 CFR 1.10(c) on the date indicated above and is addressed to:

> **BOX PCT**
> **Assistant Commissioner for Patents**
> **Washington DC 20231**

| | |
|---|---|
| Case Number: | **P00,1950** |
| Applicant(s): | **Michael Berger** |

| | |
|---|---|
| **International Application No.** | **PCT/DE99/01999** |
| **International Filing Date** | **01 July 1999** |
| **Priority Date Claimed** | **27 July 1998** |

**Title:      METHOD, ARRANGEMENT AND SET OF A PLURALITY OF ARRANGEMENTS FOR ELIMINATING AT LEAST ONE INCONSISTENCY IN A DATABASE COLLECTION CONTAINING A DATABASE AND AT LEAST ONE COPY DATABASE OF THE DATABASE**

**Enclosed are the following documents:**
International application as filed, drawings attached;
English Translation, translated drawings attached;
Annexes;
Executed Declaration;
Change of Address form for Applicants' Representative;
PTO 1390 in duplicate;
Amendment "A" prior to action and Appendix "A";
Information Disclosure Statement, PTO 1449, Search Report, 06 References;
Appointment of Associate Power of Attorney;
Drawing Additions;
Substitute Specification and Mark-Up for Substitute Specification;


Fee: $914.00
Postcard.

**(See attached envelope for Executed Assignment;
PTO 1595; $40.00 filing fee; Postcard)**

*Sharon Perkins*
_____
Signature of person mailing documents and fees

BOX PCT
IN THE UNITED STATES DESIGNATED/ELECTED OFFICE
OF THE UNITED STATES PATENT AND TRADEMARK OFFICE
UNDER THE PATENT COOPERATION TREATY--CHAPTER II

APPLICANT(S):                          MICHAEL BERGER

ATTORNEY DOCKET NO.:                   P00,1950

INTERNATIONAL APPLICATION NO:  PCT/DE99/01999

INTERNATIONAL FILING DATE:     01 JULY 1999

INVENTION:   METHOD, ARRANGEMENT AND SET OF A
             PLURALITY OF ARRANGEMENTS FOR ELIMINATING
             AT LEAST ONE INCONSISTENCY IN A  DATABASE
             COLLECTION CONTAINING A DATABASE AND AT
             LEAST ONE COPY DATABASE OF THE DATABASE


Assistant Commissioner for Patents,
Washington D.C. 20231

## AMENDMENT A PRIOR TO ACTION

Sir:

Applicants herewith amend the above-referenced PCT application, and
request entry of the Amendment prior to examination on the United States
Examination Phase.


## IN THE CLAIMS:

**On page 50:**

replace line 1 with --WHAT IS CLAIMED IS:--;

Please replace original claims 1-23 with the following rewritten claims 1-23,
referring to the mark-ups in Appendix A.

1.      (Amended) A method for a computer-aided elimination of at least one
inconsistency in a database collection containing a database and at least one copy
database of the database, comprising the steps of:

changing said database or said at least one copy database, thereby
producing an inconsistency;

allocating at least some operations which create an inconsistency to defined

1

conflict types;

allocating each conflict type a decision set which is used to indicate possible decisions which can be used to eliminate an inconsistency created by at least one operation of said respective conflict type; and

eliminating said inconsistency utilizing said decision set.

2.    (Amended) The method as claimed in claim 1, further comprising the step of eliminating additional inconsistencies.

3.    (Amended) The method as claimed in claim 1, further comprising the step of allocating each conflict type a decision set which is used to indicate possible decisions which can be used to eliminate an inconsistency created by additional operations of the respective conflict type.

4.    (Amended) The method as claimed in claim 1, wherein said database collection contains a plurality of copy databases of said database.

5.    (Amended) The method as claimed in claim 2, further comprising the step of ascertaining all inconsistencies and their dependencies on one another before said step of eliminating said inconsistency.

6.    (Amended) The method as claimed in claim 5, further comprising the step of ascertaining a conflict, an anomaly, or a pseudo-anomaly when an inconsistency is ascertained.

7.    (Amended) The method as claimed in claim 2, further comprising the step of modifying, during elimination of said inconsistencies, said decision set for at least one conflict type depending on dependencies of said inconsistencies.

2

8.    (Amended) The method as claimed in claim 2, further comprising the step of examining, after a prescribable number of eliminated inconsistencies, said database collection for further inconsistencies and their dependencies, anomalies and pseudo-anomalies.

9.    (Amended) The method as claimed in claim 1, wherein said database collection contains an object-oriented database.

10.   (Amended) The method as claimed in claim 1, further comprising the step of applying said method in a context of object-oriented software development.

11.   (Amended) The method as claimed in claim 1, further comprising the step of applying said method in a context of creating a structured electronic document.

12.   (Amended) An arrangement for eliminating at least one inconsistency in a database collection containing a database and at least one copy database of said database, which inconsistency arises on account of the database or said at least one copy database being changed, comprising:
        a processor configured to:
            allocate at least some operations which create an inconsistency to defined conflict types;
            allocate to each conflict type a decision set which is used to indicate possible decisions which can be used to eliminate an inconsistency created by at least one operation of said respective conflict type; and
            eliminate said inconsistency using said decision set.

13.   (Amended) The arrangement as claimed in claim 12, wherein said processor is configured to eliminate a plurality of inconsistencies.

3

14. (Amended) The arrangement as claimed in claim 12, wherein said processor is configured to allocate each conflict type a decision set which is used to indicate possible decisions which can be used to eliminate an inconsistency created by a plurality of operations of said respective conflict type.

15. (Amended) The arrangement as claimed in claim 12, wherein said processor is configured to operate a database collection that contains a plurality of copy databases of said database.

16. (Amended) The arrangement as claimed in claim 13, wherein said processor is configured to ascertain all inconsistencies and their dependencies on one another before said inconsistencies are eliminated.

17. (Amended) The arrangement as claimed in claim 12, wherein said processor is configured to a certain a conflict, an anomaly or a pseudo-anomaly when an inconsistency is ascertained.

18. (Amended) The arrangement as claimed in claim 13, wherein said processor is configured to modify, during elimination of said inconsistencies, a decision set for at least one conflict type depending on dependencies of said inconsistencies.

19. (Amended) The arrangement as claimed in claim 13, wherein said processor is configured to examine, after a prescribable number of eliminated inconsistencies, said database collection for further inconsistencies and their dependencies, anomalies and pseudo-anomalies.

20. (Amended) The arrangement as claimed in claim 12, wherein said processor is configured to operate on said database collection that contains an object-oriented database.

4

21.     (Amended) The arrangement as claimed in claim 12, wherein said processor is configured to operate in a context of object-oriented software development.

22.     (Amended) The arrangement as claimed in claim 12, wherein said processor is configured to operate in a context of creating a structured electronic document.

23.     (Amended) A set of a plurality of arrangements for eliminating at least one inconsistency in a database collection containing a database and at least one copy database of said database, which inconsistency arises on account of said database or said at least one copy database being changed, comprising;

a plurality of processors, wherein each arrangement has at least one processor which is configured to:

allocate at least some operations which create an inconsistency to defined conflict types;

allocate to each conflict type a decision set which is used to indicate possible decisions which can be used to eliminate an inconsistency created by at least one operation of said respective conflict type; and

eliminate said inconsistency using said decision set;

said arrangements being configured to be coupled to one another.

## REMARKS

The present Amendment revises the specification and claims to conform to United States patent practice, before examination of the present PCT application in the United States National Examination Phase. Pursuant to 37 CFR 1.125 (b), applicants have concurrently submitted a substitute specification, excluding the claims, and provided a marked-up copy. All of the changes are editorial and applicant believes no new matter is added thereby. The amendment, addition, and/or cancellation of claims is not intended to be a surrender of any of the subject matter of those claims.

Early examination on the merits is respectfully requested.

Submitted by,

_____ (Reg. No. 45,877)
Mark Bergner
Schiff Hardin & Waite
Patent Department
6600 Sears Tower
233 South Wacker Drive
Chicago, Illinois 60606-6473
(312) 258-5779
Attorneys for Applicant

6

Description

**Method, arrangement and set of a plurality of**
5    **arrangements for eliminating at least one inconsistency**
**in a database collection containing a database and at**
**least one copy database of the database**

The invention relates to a method, an arrangement and a
10   set of a plurality of arrangements for eliminating at
least one inconsistency in a database collection
containing a database and at least one copy database of
the database.

15   Such a method is disclosed in [1].

In the method disclosed in [1], computers communicate
with one another via a communication network using a
communication protocol.
20
A communication network is to be understood as meaning,
by way of example, a data network, a radio network or
else a conventional telephone network.

25   A communication protocol is to be understood as meaning
a protocol for stipulating the data format used for
communication between the computers. Such a
communication protocol is the Transport Control
Protocol/Internet Protocol (TCP/IP), for example.
30
In the method in [1], a first computer stores a
database and each further computer stores a copy of the
database, called copy database below.

35   The database and the copy databases are modified by a
respective computer during a session, i.e. the data
contained in the database or in a copy database, or the
structure of said data, is modified.

In this context, a database is to be understood as meaning a hierarchical or else object-oriented database, for example.

5   A database contains data which is stored in accordance with a prescribed structure and is interrelated. Each object, i.e. each data record within the database, is usually unambiguously identifiable by means of an identifier.

10

Changes to a copy database are sometimes made without the same change also being made in the database itself, or else vice versa.

15   If a consistent database is now meant to be created from the respective copy database and the database, then it is necessary to ascertain and eliminate any inconsistency arising as a result of the data, or the structure of said data, being added, removed or
20   changed.

An inconsistency is to be understood below as meaning any syntactical difference within a copy database and the database, i.e. any discrepancies arising in the
25   copy databases and the database between the data elements contained in the database and a copy database, the properties of said data elements and their relationships with one another.

30   [1] indicates various options for eliminating such an inconsistency.

The invention is based on the problem of specifying another method and another apparatus for eliminating
35   inconsistencies in a database collection which allows an inconsistency to be eliminated whilst saving computation time as far as possible.

The problem is solved by the method, by the arrangement and by the set of a plurality of arrangements having the features of the independent patent claims.

5   The method for the computer-aided elimination of at least one inconsistency in a database collection containing a database and at least one copy database of the database, which inconsistency arises on account of a change in the database and/or in the copy database,
10  has the following steps:

a)   at least some of the operations which can create an inconsistency are allocated to defined conflict types,

15

b)   each conflict type is allocated a decision set which is used to indicate possible decisions which can be used to eliminate an inconsistency created by at least one operation of the respective
20       conflict type,

c)   the inconsistency is eliminated using the decision set.

25  The arrangement for eliminating at least one inconsistency in a database collection containing a database and at least one copy database of the database, which inconsistency arises on account of the data in the database or in the copy database being
30  changed, has at least one processor which is set up such that the following steps can be carried out:

a)   at least some of the operations which can create an inconsistency are allocated to defined conflict
35       types,

b)   each conflict type is allocated a decision set which is used to indicate possible decisions which

can be used to eliminate an inconsistency created
by an operation of the respective conflict type,

c)  the inconsistency is eliminated using the decision
5     set.

The set of a plurality of arrangements for eliminating
at least one inconsistency in a database collection
containing a database and at least one copy database of
10  the database, which inconsistency arises on account of
the database and/or the copy database being changed,
contains a plurality of arrangements, each of which has
at least one processor which is set up such that the
following steps can be carried out:
15  a)  at least some of the operations which can create an
inconsistency  are  allocated  to  defined  conflict
types,
b)  each  conflict  type  is  allocated  a  decision  set
which is used to indicate possible decisions which
20    can be used to eliminate an inconsistency created
by  at  least  one  operation  of  the  respective
conflict type,
c)  the inconsistency is eliminated using the decision
set.
25  The arrangements can be coupled to one another.

The invention makes it possible to generically resolve
an inconsistency in a complex database.

30  Preferred developments of the invention can be found in
the dependent claims.

In   one   preferred   embodiment,   a   plurality   of
inconsistencies are eliminated.
35

Preferably, in a further embodiment, each conflict type
is allocated a decision set which is used to indicate
possible decisions which can be used to eliminate an

inconsistency created by a plurality of operations of the respective conflict type.

In addition, in one development, the database
5 collection contains a plurality of copy databases of the database.

For the purposes of simplification and hence to save computation time when eliminating an inconsistency, one
10 embodiment of the invention provides for all inconsistencies and their dependencies on one another to be ascertained before elimination.

A further saving of the computation time required to
15 eliminate a plurality of inconsistencies is achieved in a further embodiment by virtue of the decision set for at least one conflict type being modified during elimination of the inconsistencies.

In this context, the respective decision set is
20 preferably changed on the basis of dependencies of the inconsistencies.

In one preferred embodiment, after a prescribable
25 number of eliminated inconsistencies, the database collection is examined for further inconsistencies and their dependency.

In one embodiment, the database collection preferably
30 contains an object-oriented database.

The method may be used within the context of object-oriented software development or else in the context of creating a structured electronic document.
35

An illustrative embodiment of the invention is shown in the figures and is explained in more detail below.

In the figures

Figure 1   shows a flowchart showing the method steps of
           the illustrative embodiment;

5   Figure 2   shows a sketch showing computers connected to
           one another via a communication network;

Figure 3   shows a sketch of a database structure;

10   Figure 4   shows   a   tabular   overview   of   possible
           conflicts   and   decision   sets   with   decision
           options   for   eliminating   the   respective
           conflicts.

15   **Figure 2** shows a first computer 200 having a memory 202
     and a processor 203 which are connected to one another
     and to an input/output interface 201 by means of a bus
     204 in each case.

20   The input/output interface 201 is used to connect the
     first computer 200 to a screen 205, to a keyboard 206
     and to a computer mouse 207.

     In addition, the first computer 200 is connected to
25   further computers 210, 220, 230, 240 and 250 via a
     communication network 260, in the example an ISDN
     network (Integrated Services Digital Network).

     The first computer 200 stores a database 208.
30
     The further computers 210, 220, 230, 240 and 250 each
     likewise have a processor 213, 223, 233, 243 and 253
     and each have a memory 212, 222, 232, 242 and 252. The
     processor 213, 223, 233, 243 and 253 and the memory
35   212, 222, 232, 242 and 252 are each connected by means
     of a respective bus 214, 224, 234, 244 and 254 to the
     communication network 260 via an input/output interface
     211, 221, 231, 241 and 251. In addition, the further
     computers 210, 220, 230, 240 and 250 are each connected

to a screen 215, 225, 235, 245 and 255 and to a
keyboard 216, 226, 236, 246 and 256 and to a computer
mouse 217, 227, 237, 247 and 257.

5   A respective copy of the database 208, called copy
database 218, 228, 238, 248 and 258 below, is
transmitted from the first computer 200 to a respective
further computer 210, 220, 230, 240 and 250 and is
stored there in the memory 212, 222, 232, 242 and 252
10  thereof.

Once the copy databases 218, 228, 238, 248 and 258 have
been transmitted, the computers 200, 210, 220, 230, 240
and 250 interrupt communication and an independent
15  change is made among the respective computers 200, 210,
220, 230, 240 and 250, i.e. removal or addition of data
or removal or addition of dependencies of the data in a
copy database 218, 228, 238, 248 and 258 and in the
database 208.
20

Once communication between the first computer 200 and
the further computers 210, 220, 230, 240 and 250 is
resumed, a consistent database is intended to be formed
from the database 208 and the copy databases 218, 228,
25  238, 248 and 258.

For this purpose, it is necessary to establish
respective changes made in the database 208 or in the
copy databases in order thus to ascertain
30  inconsistencies between the copy databases and the
database 208 so that the inconsistencies can be
eliminated.

Independently of the syntactical structure and of the
35  dependencies of the data elements on one another, each
data element can have an arbitrary number of
properties. In this context, each property is of a
particular property type and is represented by a
current value. For all properties, it is assumed with

regard to the value ranges that all values may comprise only symbols or compiled symbols from an ASCII table (digits, numbers, letters, special characters, character chains). A series of such characters and
5   symbols is called an entry below. More complex properties are represented in the application modeling by data elements and relationships.

Three types of properties for the syntactical analysis
10  are distinguished below on the basis of the operations which can be carried out on the property:

• **Individual value:**
   A "value" as property type describes an individual
15  entry, the entry always being regarded in its entirety and also being modified as such. In this context, the "value" type property is always modified by complete replacement of the entry for the property with a new entry.
20 • **Listing:**
   A "listing" as property type describes a quantity of arbitrary entries, the entries bearing no relation to one another and, for their part, being able to represent an individual value, a listing or an
25  ordered listing. In this context, the individual entries can be added or deleted only individually. The unambiguity of the entries must be ensured if requested by the application. An example of a data structure representing this property type is a hash
30  table or an array.
   • **Ordered listing:**
   Properties of the "ordered listing" type describe a quantity of arbitrary entries, like properties of the simple listing type. In this case, however, the
35  entries are in a defined order with respect to one another, said order being stipulated by means of an index for each entry. The indices are stipulated relative to the start of the listing. An insertion operation with individual entries or with a plurality

of entries therefore always relates to an index. A
deletion operation can relate to an individual entry
with only one index or to a series of successive
entries, and hence to a start index and an end index.
5  The criterion for the order is defined by an
application program, and the observance of the order
criteria is also monitored by the latter. An example
of this property type is an indexed list containing
arbitrary entries (e.g. text document), in which each
10  line or each character corresponds to one entry.

A data element DE is to be understood as meaning a
4-tuple defined as follows:

15  **Data element**
A data element DE is a 4-tuple

DE _def_ (ID, infospace, elementtype, properties);

20  • ID is a one-to-one identifier throughout the system
• infospace ∈ MIR; where MIR is a quantity of all the
  information spaces
• elementtype ∈ ET; where ET is a quantity of all the
  data element types
25  • properties ⊆ {(name, propertytype, value):
• name ∈ MEN, value ∈ MEW, propertytype ∈ MET},
  where:
  MEN is a quantity of all the property names and the
  following applies:
30

$$\forall\ i \in \{1,...,n\};\ \forall\ k \in \{1,\ ...,m\}: name_i \neq name_k,$$

MEW is a quantity of all the property values, and
MET is a quantity of all the property types
35  {"value", "listing", "ordered listing"}.

In addition, an information space is defined as
follows:

**Information space**

An information space IR is a 3-tuple

IR _def_ (ID, irname, owner, data)

- ID is a one-to-one identifier throughout the system,
  where the following applies:
  $\forall i \in (1, ..., n); \forall k \in (1, ..., n): i \neq k:$
  $IRi.ID \neq IRk.ID;$
  where MIR is a quantity of all the Irs present in the
  system and n is the number thereof;
- irname $\in$ MIRN
  where the following applies:
  $\forall i \in (1, ..., m); \forall k \in (1, ..., m): i \neq k:$
  $Iri.irname \neq Irk.irname;$
  where MIR is the quantity of all the Irs present in
  the system and m is the number thereof;
  MIRN represents a quantity of all the possible
  information space names;
- owner = Ni where Ni $\in$ MN or Ngi where Ngi $\in$ MNG;
- data is the data which can be accessed by a user
  group and is associated with the IR.

A relationship between the data elements is
additionally defined as follows:

**Relationship between data elements**

A relationship BZ between data elements is a 3-tuple

BZ _def_ (relationshiptype, name, dataelement1,
dataelement2)

- name $\in$ MBN; where MBN is a quantity of all the
  relationship names
- relationshiptype $\in$ MBT; where MBT is a quantity of
  all the relationship types
  {"undirected", "logical", "successor", "subsup"}
- dataelement1, 2 $\in$ MDE; where MDE is a quantity of all
  the data elements.

In order to ascertain inconsistencies, each computer 200, 210, 220, 230, 240 and 250 carries a respective protocol relating to all the operations carried out on the database or on the respective copy database and
5    stores it in the form of a list.

The stored list is called the history below.

Hence, the database 208 and each copy database 218,
10   228, 238, 248 and 258 have a respective associated history.

This situation is shown in **Figure 3**. **Figure 3** shows the database 301 containing objects 302, 303, 304 and 305
15   and also a history 306 storing, as entries 307, 308, 309, change operations which have been carried out on the database 301 by the first computer 200 since communication with the further computers 210, 220, 230, 240 and 250 was interrupted. The entries 307, 308, 309
20   are likewise stored in the memory 202 of the first computer 200.

A first copy database 310 containing objects 311, 312, 313 and 314 likewise has an associated history 315 with
25   appropriate change operations 316, 317, 318. The copy database 310 is stored in the further computer 210.

A second copy database 320 containing objects 321, 322, 323 and its associated history 325 with change
30   operations 326, 327, 328 is stored in a further computer 220.

To form the consistent database, i.e. to reintegrate all the copy databases 218, 228, 238, 248 and 258 with
35   the database 208, the histories 315, 325, ... are transmitted to the first computer 200 via the communication network 260 and are stored in the memory 202 of the first computer 200.

At the start of reintegration, which is described in **Figure 1** by step 101, all the histories of the copy databases are transmitted to the first computer and are stored there (step 102).

5

In a third step (step 103), all the histories 315, 325, ... to be taken into account for reintegration are determined.

10     In addition, the following change operations are taken into account and are used to uniquely describe the inconsistencies which have arisen.

Within the context of this illustrative embodiment, the
15     following nine operations are taken into account as change operations, which are described below in the form of a pseudo-program code:

**1. Create Element:**
20     **Create Element** (R(IR), ID, Element type) → R(IR)

```
createElement(ir, id, elementtype) RETURN R(IR)
     BEGIN element := instantiate(elementtype)
           element.elementname:= id
25         R(ir):= insert(R(ir).data,element)
           R(ir):= add(R(ir).history,
           "createElement(id,elementtype)")
           return R(ir)
     END
```
30

This operation creates a data element of the data type **elementtype** with the identifier **id** in a copy database **R(ir)** within an information space **ir**, to which this operation is applied. In this context, all the
35     properties of the newly created data element receive a preset initialization value. Having been initialized, the new element is added under the specified name to the data in the copy database **R(ir)** and the executed operation is added without the information space as a

parameter into the history **R(ir).history** associated
with the copy database. In this context, the unique
identifier **id** can be created by the application sending
the operation.

5

**2. DeleteElement:**
**DeleteElement**(R(IR),ID) → R(IR)

```
deleteElement(ir,id)RETURN R(IR)
10      BEGIN element:= select (R(ir).data,id)
             R(ir):= remove(R(ir).data, element)
             R(ir):= add(R(ir).history,"deleteElement(id)")
             return R(ir)
        END
15
```

This operation deletes a data element with the name **id**
from the copy database **R(ir)** of the information space
**ir** and writes the executed operation into the history
**R(ir).history** associated with the copy database. All

20  the properties of the data element which have been
modified since the instantiation of the data element
are also lost in the process. The relationships
affecting the element are retained, however. If these
are likewise to be deleted, then the application

25  program is responsible for this.

**3. ChangeProperty:**
**ChangeProperty**(R(IR), ID, Property type, Value) →
R(IR)

30

```
changeProperty(ir, id, propname, newValue) RETURN R(IR)
        BEGIN element := select (R(ir).data,id)
             property := select(element.properties,
             propname)
35           property.value := newValue
             R(ir):= add (R(ir).history, "changeProperty
             (id, propname, newValue)")
             return R(ir)
        END
```

- 14 -

This operation sets the value of the property **propname** of the data element with the identifier **id** to the value **newValue** in the copy database **R(ir)** of the information space **ir** and writes the executed operation into the history **R(ir).history** associated with the copy database.

**4. ChangePropertyAdd:**

**ChangePropertyAdd**(R(IR),ID,Property type,Entry)→R(IR)

```
changePropertyAdd(ir,id,propname,newEntry) RETURN R(IR)
      BEGIN element := select (R(ir).data, id)
            property := select (element.properties,
            propname)
            add (property.value, newEntry)
            R(ir) := add(R(ir).history,
            "changePropertyAdd (id, propname,
            newEntry)")
            return R(ir)
      END
```

This operation is for properties of the "listing" type. The operation adds a new entry **newEntry** to the copy database **R(ir)** of the information space **ir** in the value of the property **propname** of the data element with the identifier **id** at the end of the listing. The executed operation is then stored in the history **R(ir).history** associated with the copy database.

**5. ChangePropertyDel:**

**ChangePropertyDel**(R(IR), ID, Property type, Index, Entry) → R(IR)

```
changePropertyDel(ir, id, propname, index, oldEntry)
                  RETURN R(IR)
      BEGIN element := select (R(ir).data, id)
            property := select (element.properties,
            propname)
            del (property.value, oldEntry)
```

```
            R(ir) := add(R(ir).history,
            "changePropertyDel(id, propname, index,
            oldEntry)")
            return R(ir)
5       END
```

This operation is for properties of the "listing" type.
The operation deletes the first entry **oldEntry**
occurring in the listing in the copy database **R(ir)** of
10  the information space **ir** in the value of the property
**propname** of the data element with the identifier **id**.
The executed operation is then stored in the history
**R(ir).history** associated with the copy database.

15  **6. ChangePropertyInsert:**
**ChangePropertyInsert**(R(IR), ID, Property type, Index,
                      Number, Entries) → R(IR)

**changePropertyInsert**(ir, id, propname, index, number,
20                  newEntries) RETURN R(IR)
```
        BEGIN element := select (R(ir).data, id)
              property := select (element.properties,
              propname)
              for (i=0, i < number, i++){
25            incrIndex (property.value.entries,ind >=
              index)
              insert (property.value.index, newEntries.
              (number -i))}
              R(ir) := add (R(ir).history,
30            "changePropertyInsert (id, propname, index,
              number, newEntries)")
              return R(ir)
        END
```

35  A property to which this operation can be applied is of
the "ordered listing" type. This operation inserts a
number number of new entries newEntries.i in the copy
database **R(ir)** of the information space **ir** in the value
of the property **propname** of the data element with the

identifier **id** from the position Index in the ordered
listing Property.Value. For all the entries in the
ordered listing **property.Value** having an identical
index to or a larger index than **index**, **index** is
5   increased by the value **number**. The executed operation
is then stored in the history **R(ir).history** associated
with the respective copy database.


**7. ChangePropertyRemove:**

10  **ChangePropertyRemove**(R(IR), ID, Property type, Index,
                       Number, Entries) → R(IR)


**changePropertyRemove**(ir, id, propname, index, number,
                      oldEntries) RETURN R(IR)
15       BEGIN element := select (R(ir).data,id)
                property := select (element.properties,
                propname)
                for (i=0, i < number, i++){
                remove (property.value.index, oldEntries.
20              (i+1))
                decrIndex (property.value.entries, ind >
                index)}
                R(ir) := add(R(ir).history,
                "changePropertyRemove(id, propname, index,
25              number, oldEntries)")
                return R(ir)
         END


A property to which this operation can be applied is of
30  the "ordered listing" type. This operation deletes the
entries **oldEntries** in the copy database **R(ir)** of the
information space **ir** in the value of the property
**propname** of the data element with the identifier **id**
from the position Index in the ordered listing
35  **Property.Value**. All entries with a larger index than
**(index + number)** have their index reduced by the value
**number**. The executed operation is then stored in the
history **R(ir).history** associated with the respective
copy database.

**8. CreateRelationship:**
**CreateRelationship**(R(IR), Name, Relationship type,
ID1, ID2)→R(IR)

```
 5   createRelationship(ir, name, reltype, fromid, toid,
                       toidir) RETURN R(IR)
         BEGIN relationship := instantiate (reltype)
               relationship.name := name
               relationship.dataelement1 := fromid
10             relationship.dataelement2 := toid
               relationship.dataelement2.ir := toidir
               R(ir) := insert (R(ir).data, relationship)
               R(ir) := add (R(ir).history,
               "createRelationship (name, reltype, fromid,
15             toid, toidir)")
               return R(ir)
         END
```

This operation creates a relationship of the **reltype**
20 type between the data elements with the identifiers
**fromid** and **toid** under the name **name** and adds the new
relationship to the data in the copy database **R(ir)** of
the information space **ir**. The executed operation is
then stored in the history **R(ir).history** associated
25 with the copy database. It is assumed for all
relationships that, for each relationship name
allocated, there is only a single relationship of the
same type between two data elements. If a plurality of
relationships of the same type are necessary under the
30 same name between two data elements, then identifiers
need to be introduced for relationships as well.
However, the assumption made is sufficient for the
majority of applications. The information space of the
target data element need be specified only in the case
35 of a logically externally directed relationship.

**9. DeleteRelationship:**

**DeleteRelationship**(R(IR), Name, Relationship type, ID1,
                    ID2) →R(IR)

**deleteRelationship**(ir,     name,     reltype,     fromid,
5   toid.toidir)

                        RETURN R(IR)
        BEGIN relationship := select (R(ir).data, name,
                    reltype, fromid, toid)
            R(ir) := remove (R(ir).data, relationship)
10          R(ir) := add (R(ir).history,
            "deleteRelationship (name, reltype, fromid,
            toid.toidir)")
            return R(ir)
        END
15

This operation deletes a relationship of the **reltype**
type between the data elements with the identifiers
**fromid** and **toid** under the name **name** from the copy
database **R(ir)** of the information space **ir**. The
20 executed operation is then stored in the history
**R(ir).history** associated with the copy database. The
information space of the target data element need be
specified only for the relationships of the logically
externally directed type.
25

In a further step, all conflicts, dependencies,
anomalies, pseudo-anomalies and restrictions by
dependencies are recognized (step 103).

30 A conflict is to be understood as meaning the smallest
decidable quantity of operations which arise only on
one side in syntactical terms, uniquely describe an
inconsistency and can be meaningfully presented to a
user or to the system and eliminated (decided) by the
35 latter.

Each conflict can be recognized as a whole and can be
resolved by a single decision during reintegration.

Potential conflicts are subsequently defined on the basis of the data structure and the operations occurring in the history.

5    A distinction is drawn between harmless conflicts and critical conflicts.

Harmless conflicts (HK) contain only operations which describe modifications on one copy database. In this
10   case, there is therefore just one user desiring and making a change to the part of the data structure or the copy database or else the database itself. The operations carried out are thus complemented. Depending on which copy database is to be allocated the operation
15   or operations, when the copy databases of users A and B are present, harmless conflicts with operations on the copy database of a first user A can be referred to as HKA and harmless conflicts with operations on the copy database of a second user B can be referred to as HKB,
20   on a general basis.

By contrast, critical conflicts (KK) contain changes on both sides to the same part of the data structure and represent contrary views of the users about the
25   ultimate state of particular data within the database and copy databases. In this context, a critical conflict can also be defined by different operations on two copy databases. In such cases, a distinction is drawn between a critical conflict KKA and a critical
30   conflict KKB.

Formally, a conflict is defined as follows:

**Conflict:**
35   A conflict K between two histories EHA and EHB and a common history GH is a 6-tuple
K(EHA, EHB, GH) _def_

        (id, ktype, operationsEHA, operationsEHB,

operationsGH, decisionrestr);

- id is a one-to-one identifier throughout the system (see also the definition of a data element)
5 - ktype ∈ {HK1A, ..., HK11A, HK1B, ..., HK11B, KK1, KK2,

        KK3A, KK3B, KK4, KK5A, KK5B, KK6, KK7, KK8A, KK8B}

- operationsEHA ∈ EHA.operations;
10 - operationsEHB ∈ EHB.operations;
- operationsGH ∈ GH.operations;
- decisionrestr ⊆ MENTktype; where MENTktype is a quantity of all the possible decisions for a conflict of the type ktype.

15 The conflicts are shown in **Figure 4**.

**1. First harmless conflict HK1:**
HK1 = (createElement/ -- )
20 There is a creation operation for a data element createElement(id, elementtype) in one history only.
HK1A = createElement(id, elementtype) ∈ EHA ∨
HK1B = createElement(id, elementtype) ∈ EHB.

25 **2. Second harmless conflict HK2:**
HK2 = (deleteElement/ -- )
There is a deletion operation for a data element deleteelement(id, elementtype) in one history only.
HK2A = deleteElement(id, elementtype) ∈ EHA ∨
30 HK2B = deleteElement(id, elementtype) ∈ EHB.

**3. Third harmless conflict HK3:**
HK3 = (createRelationship/ -- )
There is a creation operation for a relationship
35 createRelationship (reltype, rname, id1, id2) in one history only.
HK3A = createRelationship(reltype, rname, id1, id2) ∈ EHA ∨
HK3B = createRelationship(reltype, rname, id1, id2) ∈ EHB.

**4. Fourth harmless conflict HK4:**

HK4 = (deleteRelationship/ -- )

There is a deletion operation for a relationship deleteRelationship (reltype, rname, id1, id2) in one history only.

HK4A = deleteRelationship(reltype,rname,id1,id2) ∈ EHA ∨
HK4B = deleteRelationship(reltype,rname,id1,id2) ∈ EHB.

**5. Fifth harmless conflict HK5:**

HK5 = (deleteRelationship12, createRelationship 13/ -- )

There is a deletion operation for a relationship deleteRelationship(reltype, rname id1, id2) and a subsequent creation operation for the relationship creationRelationship(reltype, rname, id1, id3) from the same source data element to another target data element in one history only.

HK5A = [deleteRelationship(reltype,rname,id1,id2),
        createRelationship(reltype,rname,id1,id3)] ∈ EHA ∨
HK5B = [deleteRelationship(reltype,rname,id1,id2),
        createRelationship(reltype,rname,id1,id3)] ∈ EHB.

**6. Sixth harmless conflict HK6:**

HK6 = (changeProperty/ -- )

There is a change operation changeProperty(id, name, valuenew, valueold) for a property of the "Value" type in one history only.

HK6A = changeProperty(id,name,value1,value0) ∈ EHA ∨
HK6B = changeProperty(id,name,value1,value0) ∈ EHB.

**7. Seventh harmless conflict HK7:**

HK7 = (n x changePropertyAdd/ -- )

There are n (n is an element of the natural numbers and n > 0) insertion operations changePropertyAdd(id, name, entry) with the same entry **entry** for the same property of the "listing" type for a data element in one history, with there being no deletion operation with the same entry for the same property of the data element in another history. The inconsistency described consists in the fact that there are n entries of the

type **entry** more in the copy database with the creation
operations in the history associated with the copy
database than in the other copy database.
HK7A = n times changePropertyAdd(id,name,entry) ∈ EHA ∨
5      HK7B = n times changePropertyAdd(id,name,entry) ∈ EHB.


**8. Eighth harmless conflict HK8:**
HK8 = (n x changePropertyDel/ -- )
There are n (n is an element of the natural numbers)
10    deletion operations changePropertyDel(id, name, entry)
with the same entry **entry** for a property of the
"listing" type for a data element in one history, with
there being no insertion operation with the same entry
for the property of the data element in another
15    history. The inconsistency described consists in the
fact that the n entries of the type **entry** are present
to a lesser extent in the copy database whose history
contains the deletion operations than in the other copy
database.
20    HK8A = n times changePropertyDel(id,name,entry) ∈ EHA ∨
HK8B = n times changePropertyDel(id,name,entry) ∈ EHB.


**9. Ninth harmless conflict HK9:**
HK9 = (changePropertyInsert/ -- )
There is an insertion operation
25    changePropertyInsert(id, name, index1, 1, entry1) for an
index with an individual entry for a property of the
"ordered listing" type for a data element in one history
only, with another history containing no insertion
30    operations with another entry for a checkable identical
index for the same property of the data element.
HK9A = changePropertyInsert(id,name,index,1,entry)
       ∈ EHA ∨
HK9B = changePropertyInsert(id,name,index,1,entry)
35        ∈ EHB.


**10. Tenth harmless conflict HK10:**
HK10 = (changePropertyRemove/ -- )
There is a deletion operation

changePropertyRemove(id, name, index1, 1, entry1) for an
index with an individual entry for a property of the
"ordered listing" type for a data element in one history
only.

5   HK10A = changePropertyIRemove(id,name,index,1,entry)
              ∈ EHA ∨

    HK10B = changePropertyIRemove(id,name,index,1,entry)
              ∈ EHB.


10  **11. Eleventh harmless conflict HK11:**

    HK11 = (changePropertyRemove,changePropertyInsert/--)
    There is a deletion operation
    changePropertyRemove(id, name, index1, 1, entry1) for an
    index **index** with an individual entry for a property of
15  the "ordered listing" type for a data element, and a
    subsequent creation operation
    changePropertyInsert (id, name, index1, 1, entry2) for an
    entry for the same index for the same property of the
    same data element in one history only.

20  HK11A = [changePropertyIRemove(id,name,index,1,entry1),
                changePropertyIInsert(id,name,index,1,entry2)]
                ∈ EHA ∨

    HK11B = [changePropertyIRemove(id,name,index,1,entry1),
                changePropertyIInsert(id,name,index,1,entry2)]
25              ∈ EHB.


    The following text gives an overview of critical
    conflicts (KK), i.e. operations in a plurality of
    histories:

30
    **1. First critical conflict KK1:**

    KK1 = (createRelationship12/ createRelationship13)
    There is a creation for a relationship
    createRelationship(reltype, rname, id1, id2) in one
35  history, with a creation
    createRelationship(reltype, rname, id1, id3) for the same
    relationship (reltype, rname), starting from the same
    source data element but to another target data element,
    existing in another history.

KK1 = createRelationship(reltype,rname,idl,id2) ∈ EHA ∧
     createRelationship(reltype,rname,idl,id3) ∈ EHB.


**2. Second critical conflict KK2:**

5  KK2 = (deleteRelationship12, createRelationship13 /
     deleteRelationship12, createRelationship14)

The different change in a relationship can be recognized
in the histories like the first critical conflict KK1. In
addition, however, a common history GH contains a

10 deletion deleteRelationship(reltype, rname, idl, id2) for
the common relationship (rname, reltype) from the same
source data element, but to another target data element.

KK2 = deleteRelationship(reltype,rname,idl,id2) ∈ GH ∧
     createRelationship(reltype,rname,idl,id3) ∈ EHA ∧

15      createRelationship(reltype,rname,idl,id4) ∈ EHB.


**3. Third critical conflict KK3:**

KK3 = (deleteRelationship12, createRelationship13/
     deleteRelationship12)

20 A relationship's modification on one side and deletion on
the other side can be recognized in the histories, like
the third harmless conflict HK3, by an operation
createRelationship(reltype, rname, idl, id3). In
addition, however, the common history contains a deletion

25 deleteRelationship (reltype, rname, idl, id2) for the
common relationship (rname, reltype) from the same source
data element but to the last common target data element.

KK3A = deleteRelationship(reltype,rname,idl,id2) ∈ GH ∧
     createRelationship(reltype,rname,idl,id3) ∈ EHA;

30 KK3B = deleteRelationship(reltype,rname,idl,id2) ∈ GH ∧
     createRelationship(reltype,rname,idl,id3) ∈ EHB.


**4. Fourth critical conflict KK4:**

KK4 = (changeProperty/ changeProperty)

35 There is a change operation
changeProperty (id, name, valuenew1, valueold) for a
property of the "value" type for a data element in one
history, with another change operation existing in

another history for the same property of the data element.

KK4 = changeProperty(id,name,valuenew1, valueold) ∈ EHA ∧
changeProperty(id,name,valuenew2, valueold) ∈ EHB.

5

**5. Fifth critical conflict KK5:**

KK5 = (n changePropertyAdd/ m changePropertyDel)

There are n (n is an element of the natural numbers) identical operations of the type

10  changePropertyAdd(id, name, entry) with the same entry for a property of the "listing" type for a data element in one history, with another history m (m is an element of the natural numbers) containing identical operations of the type

15  changePropertyDel(id, name, entry) with the same entry for the same property of the data element. The inconsistency described consists in the fact that the copy database with the creation operations in the history associated with the copy database contains n + m

20  identical entries entry more than the other copy database. To be able to make an exact statement about the occurrence of the operations, a distinction is drawn for the fifth critical conflict KK5 between a fifth critical conflict of first type KK5A and a fifth critical conflict

25  of second type KK5B. In this case, the allocation is made using the creation operations.

KK5A = (n changePropertyAdd(id,name,entry) ∈ EHA ∧
m changePropertyDel(id,name,entry) ∈ EHB ∨

KK5B = (m changePropertyDel(id,name,entry) ∈ EHA;

30      n changePropertyAdd(id,name,entry) ∈ EHB.

**6. Sixth critical conflict KK6:**

KK6 = (changePropertyInsert / changePropertyInsert)

There is an insertion operation

35  changePropertyInsert(id, name, index1, 1, entry1) for an index with an individual entry for a property of the "ordered listing" type for a data element in one history, with another history containing an insertion

operation with another entry for the same (this can be
checked) index for the property of the data element.
KK6 = changePropertyInsert(id,name,index1,1,entry1)
   ∈ EHA ∧
5  changePropertyInsert(id,name,index2,1,entry2)
   ∈ EHB.


**7. Seventh critical conflict KK7:**

KK7 = (changePropRemove, changePropInsert /
10  changePropRemove, changePropInsert)

The different change on the two sides in an individual
entry on a common index **index** for a property of the
"ordered listing" type for a data element can be
recognized in the histories like the sixth critical
15 conflict KK6. In addition, however, the common history
contains a changePropertyRemove(id, name, index1, 1,
entry1) operation for the last common entry on the same
(this can be checked) index.

KK7 = changePropertyRemove(id,name,index1,1,entry1)
20   ∈ GH ∧
  changePropertyInsert(id,name,index2,1,entry2)
   ∈ EHA ∧
  changePropertyInsert(id,name,index3,1,entry3)
   ∈ EHB.
25

**8. Eighth critical conflict KK8:**

KK8 = (changePropRemove, changePropInsert /
changePropRemove)

The change, on one side, and deletion, on the other
30 side, in/of an individual entry on a common index **index**
for a property of the "ordered listing" type for a data
element in the histories can be recognized like a tenth
harmless conflict HK10. In addition, however, the
common history contains a
35 changePropertyRemove(id, name, index1, 1, entry1)
operation for the last common entry on the same (this
can be checked) index.

KK8A = changePropertyRemove(id,name,index1,1,entry1)
   ∈ GH ∧

changePropertyInsert(id,name,index2,1,entry2)

    ∈ EHA;

KK8B = changePropertyRemove(id,name,index1,1,entry1)

    ∈ GH ∧

changePropertyInsert(id,name,index2,1,entry2)

    ∈ EHB.


The operations stored in the histories describe the autonomously modified data areas directly and are used to describe and to eliminate (described below) the inconsistencies.


To recognize the inconsistencies, two respective histories are compared with one another.


The inconsistencies are recognized at the start of the method, before the actual reintegration.


The search for the existing inconsistencies in the copy databases by searching for conflict operations is carried out in the three steps described below.


• A first step makes a pass through the two histories to be compared with one another in the copy databases and in the database, which are to be matched to one another. All the operations in the histories are allocated, separately on each side, to a respective one of the new operation collections (createElement operations, deleteElement operations, createRelationship operations, deleteRelationship operations, changeProperty operations, changePropertyAdd operations, changePropertyDel operations, changePropertyInsert operations and changePropertyRemove operations) described above.


• In a second step, a respective conflict register KR is started for each conflict type described above HK1A, ..., HK11A, HK1B, ..., HK11B, KK1, KK2, KK3A, KK3B, KK4, KK5A, KK5B, KK6, KK7, KK8A, KK8B. In this

context, it is ensured that all conflicts for which
operations from the two histories contribute to the
respective conflict are not recognized twice and
stored in the respective conflict register KR twice.

5    Accordingly, the operation collections just formed
are searched through on the basis of the definitions
of the conflict types, as described above, starting
with the first harmless conflict HK1A in the history
of the first user A. If a conflict has been

10   ascertained, the conflict is stored in the conflict
register KR for the appropriate conflict type, for
example the first harmless conflict HK1 is stored in
the copy database of the first user A in a conflict
register KR_HK1A.

15
• If the search and storage of the conflicts has taken
place, the operation collections created at the start
are deleted again in a third step. The subsequent
elimination of the inconsistencies is based on the

20   conflicts stored in the conflict registers KA and on
the operations of said conflicts.

A conflict register KR is defined as follows:

25   **Conflict register KR:**
A conflict register KR for two histories EHA and EHB
and a common history GH is a 2-tuple

$KR(EH_A, EH_B, GH) \underline{def} (crtype, conflictids)$

30
• crtype $\in$ {KA_HK1A, ..., KA_HK11A, KA_HK1B, ...,
  KA_HK11B, KA_KK1, KA_KK2, KA_KK3A, KA_KK3B, KA_KK4,
  KA_KK5A, KA_KK5B, KA_KK6, KA_KK7, KA_KK8A, KA_KK8B}
• conflictids are identifiers for all the conflicts

35   K(EHA, EHB, GH) associated with the conflict register
  KR, where K.type can be allocated to the respective
  conflict array type KR.crtype.
An anomaly is present when two data elements exist in
the two copy databases before and after the division

and the latter are connected after the division by a
directed relationship of the same type reltype and with
the same name rname, but with source data element and
target data element reversed. During the reintegration,
5   at least one of these relationships must be rejected or
the two must be modified.

A directed relationship is to be understood as meaning
a relationship which is directed from a target data
10  element to a source data element.

An anomaly is defined as follows:

**Anomaly:**
15  An anomaly AM between two conflicts K1(EHA, EHB, GH)
and K1(EHA, EHB, GH) is a 4-tuple

AM(K1, K2) _def_ (id, amtype, cid1, cid2)

20  &bull; id is a one-to-one identifier throughout the system
   (see also definition of a data element)
  &bull; amtype $\in$ {Anomaly1a, ..., Anomaly16AB}
  &bull; cid1 = K1.id
  &bull; cid2 = K2.id.
25

An anomaly register AMR for two histories is defined as
follows:

**Anomaly register AMR:**
30  An anomaly register AMR for two histories EHA and EHB
and a common history GH is a 1-tuple

AMR (EHA, EHB, GH) _def_ (anomalyids)

35  &bull; anomalyids are the identifiers for all anomalies
   between the histories EHA and EHB and the common
   history GH.
  Pseudo-anomalies describe situations in which the
  occurrence of an anomaly from conflicts which are

present can be prevented only through targeted minimization of the decision options for the conflicts.

A pseudo-anomaly is defined as illustrated below:

5

**Pseudo-anomaly PAM:**

A pseudo-anomaly PAM between two conflicts K1(EHA, EHB, GH) and K2(EHA, EHB, GH) is a 4-tuple

10 PAM(K1, K2) *def* (id, pamtype, cid1, cid2)

- id is a one-to-one identifier throughout the system (see also definition of a data element)
- pamtype ∈ {pseudo-Anomaly1A, ..., pseudo-Anomaly32AB}
15 - cid1 = K1.id
- cid2 = K2.id.

A pseudo-anomaly register PAMR is defined as follows:

20 **Pseudo-anomaly register PAMR:**

A pseudo-anomaly register PAMR for two histories EHA and EHB and a common history GH is a 1-tuple

PAMR(EHA, EHB, GH) *def* (pseudo-anomalyids)

25

- Pseudo-anomalyids are the identifiers for all pseudo-anomalies between the histories EHA and EHB and the common history GH.

30 After the conflicts have been ascertained, each ascertained conflict is resolved by means of a respective individual decision. The conflict resolution process thus comprises a sequence of conflict resolution decisions.

35

The conflict resolution is denoted in **Figure 1** by step 104.
In principle, there are various decision options:
a)Adopt the conflict operation(s)

b) Reject the conflict operation(s)

c) Partly adopt, partly reject the conflict operation(s)

d) Reject the conflict operation(s), adopt new created operation(s).

5

The individual conflict types are allocated a decision set ES, the decision set ES containing possible decisions which can be used to eliminate an inconsistency created by an operation of the respective

10   conflict type, to which a respective decision set ES is allocated.

**Figure 4** compiles all the decision sets which are allocated to a respective conflict.

15

Each row of the table shown in **Figure 4** shows a possible decision option E1, E2, E3a, E3b, E4, E5a, E5b, E6.

20   In each case, an x in a field denotes that the conflict shown in the respective column can be resolved by a decision option which is shown in the respective row.

The following text gives an overview of the possible

25   decision options:

A first decision option E1 describes the adoption of a conflict operation or of a plurality of conflict operations.

30

A conflict operation describes all the data operations associated with a conflict. Adopt is understood to mean that the conflict operations are executed in the copy database in which they have not yet been carried out.

35

A second decision option E2 describes the rejection of a conflict operation or a plurality of conflict operations.

A third decision option E3 describes the adoption of one or more conflict operation(s) in one copy database and the rejection of the conflict operation(s) in the other copy database.

5

For the third decision option E3, a detail decision is provided which defines which of the conflict operations present in the histories of different copy databases of the users A and B are to be adopted and which are to be
10   rejected.

These decision options are denoted as a first part E3a of the third decision option E3 and as a second part E3b of the third decision option E3. The first part E3a
15   of the third decision option E3 describes the adoption of the conflict operation(s) in the copy database of the first user A and the rejection of the conflict operation(s) in the copy database of the second user B. The second part E3b of the third decision option E3
20   describes the adoption of the conflict operation(s) of the copy database of the second user B and the rejection of the conflict operation(s) of the copy database of the first user A.

25   As **Figure 4** shows, a first decision set ES1, which is allocated to the first harmless conflict HK1, describes the first decision option E1 and the second decision option E2 to eliminate the first harmless conflict HK1.

30   A second decision set ES2 is allocated to the second harmless conflict HK2 and again contains the first decision option E1 and the second decision option E2 to eliminate the second harmless conflict HK2.

35   If the previous resolution options through the adoption or rejection of available conflict operations do not correspond to the target ideas of the users with regard to the ultimate reintegrated database, i.e. the users A and B cannot agree on one state described by the

conflicts, then there is the option of adopting an intermediate solution or the option of selecting and adopting new operations not contained in the decision set. The two options will be illustrated below.

5

For a conflict with a number n of identical operations, defining the conflict, from a set of data operations in only one history (conflicts of the type HK7 II and HK8 II), there are generally selection options for

10 intermediate states, as described below.

For the seventh harmless conflict HK7 with n = 1, HK7 I is written below, and HK7 II is written below for the seventh harmless conflict HK7 with n > 1. For the

15 eighth harmless conflict HK1 with n = 1, HK8 I is written below, and HK8 II is written below for the eighth harmless conflict HK8 with n > 1.

In this context, a fourth decision option E4 describes

20 a partial adoption and partial rejection of the conflict operations.

For the fourth decision option E4, a more precise statement is provided which defines how many of the

25 conflict operations arising on one side are to be adopted and how many are to be rejected. For a number of n operations (n is the element of the natural numbers), the decision options extend from adoption of one operation and rejection of n-1 operations up to

30 adoption of n-1 operations and rejection of one operation. In this case, an adequate decision option is to define the number k (0 < k < n) of adopted operations. The number of rejected operations is then calculated from n - k. The fourth decision option E4

35 can thus be specialized as follows: the fourth decision option E4 describes the number of adopted conflict operations k.

In addition, the fifth critical conflict KK5 with
n = m = 1 is denoted by KK5 I below and the fifth
critical conflict KK5 with n > 1 and m > 1 is denoted
by KK5 II below.

5

For a fifth critical conflict KK5 II with a number n of
identical operations changePropertyAdd, defining the
conflict, in one copy database and a number m of
identical operations changePropertyDel, defining the
10 conflict, in another copy database, selection of the
intermediate states is more difficult.

Since the operations defining the conflict in one
history and the operations involved in the conflict in
15 the other history cancel one another out, the same end
result can be obtained by means of different decision
options.

Thus, resetting a changePropertyAdd operation and
20 adopting a changePropertyAdd operation in one copy
database in conjunction with resetting a
changePropertyDel operation in another copy database
can obtain the same result as adopting two
changePropertyAdd operations in one copy database and a
25 changePropertyDel operation in the other copy database.
So that all decision options between the extremes of
rejecting the operations in one copy database and
adopting the operations in the other copy database
(third decision option E3a/E3b) are provided, but
30 various decision options are at the same time prevented
from supplying an identical result, there is the
solution option of the fifth decision option E5, as
illustrated below.

35 The fifth decision option E5 can create the last common
state between the copy databases by rejecting all
operations and permits all other final states from the
combinations of the operations, but without the
conflict states present above.

The fifth decision option E5 thus describes the partial adoption and partial rejection of the conflict operations in one copy database and the rejection of the conflict operations in another copy database.

For the fifth decision option E5, the decision sub-options are necessary, which firstly define which copy database is affected by the partial adoption and partial rejection and which database is affected by the total rejection, and secondly how many of the conflict operations are to be adopted in the case of partial adoption and how many are to be rejected. In this case, the number of operations during partial adoption and partial rejection can be defined as in the case of the fourth decision option E4 using the sole definition of the number of adopted operations. The number of adopted operations in a first copy database of the first user A is in this case denoted by i, and the number of adopted operations in the copy database of the second user B is denoted by k. Thus, detail decisions for the fifth decision option E5 are:

First part E5a of the fifth decision option E5:
Number of adopted conflict operations for the copy database of the first user A:
$(1 < i < n)$ if a changePropAdd operation is involved,
$(1 < k < m)$ if a changePropDel operation is involved,
and rejection of all conflict operations for the copy database of the second user B.

Second part E5b of the fifth decision option E5:
Number of adopted conflict operations for the copy database of the second user B:
$(1 < i < n)$ if a changePropAdd operation is involved,
$(1 < k < m)$ if a changePropDel operation is involved,
and rejection of all conflict operations for the copy database of the first user A.

A selection option for creating a new state for the
conflict will be defined by the decision options below.

5 In general, the option of creating and selecting a
state which differs from the two available versions is
provided for all conflicts apart from for the conflicts
of type HK1, HK2, HK4, HK10.

10 For creating a new state, it is necessary to create a
common starting position, i.e. the relevant
operation(s) must be rejected and the two copy
databases must be made consistent in terms of the data
structure affected by the conflict. This rejection of
the operations is not necessary for operations of the
15 changeProperty() type (HK6, HK4) which overwrite one
another, because the operation created with the new
state overwrites the old operations directly.

For a conflict with an operation defining the conflict
20 from the set of data operations (conflicts of the types
HK3, HK6, HK7 I, HK8 I and HK9), for a conflict with a
plurality of operations defining the conflict from the
set of data operations in the case of just one copy
database (conflicts of the types HK5, HK7 II, HK8 II,
25 HK11) and for a conflict with at least one operation
defining the conflict from the set of data operations
in the case of the two copy databases (conflicts of the
types KK1, ..., KK8), there is the following resolution
option, which is called the sixth decision option E6.

30

The sixth decision option E6 describes the rejection of
the conflict operation(s) and selection of new
operation(s).

35 In this context, to change a relationship (KK2) on both
sides or to change an entry in an ordered listing (KK7)
on both sides, the rejection, in contrast to the second
decision option E2 (rejection of all operations),
relates only to the creator operations for the new

relationships or for the new entries. The common
deletion operation for the old relationship or for the
old entry remains unaffected. For the third critical
conflict KK3 and the eighth critical conflict KK8, the
5 sixth decision option E6 has the option only of
changing the createRelationship or the
changePropertyInsert operation. The common deletion
operation remains unaffected.

10 For the definition of a new state for a conflict of
type HK7, HK8 or KK5, the number of
changePropertyAdd operations and
changePropertyDel operations is likewise described with
i and k
15 (i if changePropAdd operations are involved and
k if changePropDel operations are involved).

For the selection of a new state and the creation of
the operations necessary for this, interactions on the
20 surface of the application program are customary. If
the option exists of selecting a new state using only a
complex interaction and if the operations created by
the interaction also affect other, unresolved
conflicts, then there is the option
25 a) of using the decision set to resolve the other
   affected conflicts as well, or
   b) of shifting creation of the new state to a later
   instant, i.e. after reintegration and during the
   coupled further work.
30

If, in accordance with a), other affected conflicts are
likewise to be resolved, then it is first necessary to
reject the operations of the conflict set if conflicts
of the type HK6 and KK4 are involved and the operations
35 are not operations which overwrite one another.

**Figure 4** shows the decision sets ES1, ES2, ES3, ES4,
ES5, ES6, ES7, ES8, ES9, ES10, ES11, ES12, ES13, ES14,

ES15, ES16, ES17, ES18, ES19, ES20, ES21, ES22, which are allocated to the respective conflicts.

The sixth harmless conflict HK6 is allocated a sixth
5    decision set ES6, containing the first decision option
E1, the second decision option E2 and the sixth
decision option E6.

A twelfth decision set ES12 is allocated to the first
10   critical conflict KK1. The twelfth decision set ES12
contains four possible decisions, the second decision
option E2, the first part E3a of the third decision
option E3, the second part E3b of the third decision
option E3 and the sixth decision option E6.
15
The further decision sets are shown in **Figure 4** and
will be illustrated below for purposes of
simplification with the aid of the following list:

20   • The first harmless conflict HK1, the second harmless
conflict HK2, the fourth harmless conflict HK4 and
the tenth harmless conflict HK10 are allocated
respective decision sets comprising the first
decision option E1 and the second decision option E2.
25   • The third harmless conflict HK3, the fifth harmless
conflict HK5, the sixth harmless conflict HK6, the
first type HK7 I of the seventh harmless conflict
HK7, the first type HK8 I of the eighth harmless
conflict HK8, the ninth harmless conflict HK9 and the
30   eleventh harmless conflict HK11 are allocated
respective decision sets containing the first
decision option E1, the second decision option E2 and
the sixth decision option E6.
• The second type HK7 II of the seventh harmless
35   conflict HK7, the second type HK8 II of the eighth
harmless conflict HK8, the first critical conflict
KK1, the second critical conflict KK2, the third
critical conflict KK3, the fourth critical conflict
KK4, the first type KK5 I of the fifth critical

conflict KK5, the sixth critical conflict KK6, the seventh critical conflict KK7 and the critical conflict KK8 are allocated respective decision sets containing the second decision option E2, the first
5 part E3a of the third decision option E3, the second part E3b of the third decision option E3 and the sixth decision option E6.

- The second type KK5 II of the fifth critical conflict KK5 is allocated a decision set having six possible
10 decisions, the second decision option E2, the first part E3a of the third decision option E3, the second part E3b of the third decision option E3, the first part E5a of the fifth decision option E5, the second part E5b of the fifth decision option E5 and the
15 sixth decision option E6.

- The second type HK7 II of the seventh harmless conflict HK7 and the second type HK8 II of the eighth harmless conflict HK8 are each allocated a decision set having four possible decisions, the first
20 decision option E1, the second decision option E2, the fourth decision option E4 and the sixth decision option E6.

**Restrictions on decision options**
25

It should be noted that the correct execution of individual decisions is dependent on the presence of a data element or of a plurality of data elements in the two copy databases.
30

By way of example, for adoption in the case of the third harmless conflict HK3A, the two data elements denoted by means of their identifiers in the relationship operation in conflict must also be present
35 in the copy database of the user B. If one of the data elements is missing, or indeed if both are missing, then this decision about adoption of the operation is not possible.

It is thus evident that dependencies may exist between individual conflicts.

A dependency of a conflict on conflicts of the type HK1A, HK1B, HK2A or HK2B in terms of its decision options is defined as follows:

**Dependent conflict:**
A conflict is dependent on the first harmless conflict HK1 or on the second harmless conflict HK2 if its decision options are restricted by the presence of a first harmless conflict HK1 or of a second harmless conflict HK2.

A dependency AK of a conflict is defined as follows:

**Dependency AK of a conflict:**
A dependency AK of a conflict K1(EHA, EHB, GH) on a conflict K1(EHA, EHB, GH) is a 4-tuple

AK (K1, K2) $\underline{def}$ (id, ctype, cid1, cid2)

- id is a one-to-one identifier throughout the system (see also definition of a data element)
- ctype $\in$ {HK1A, HK1B, HK2A, HK2B}
- cid1 = K1.id
- cid2 = K2.id.

A dependency register AKR is defined as follows:

**Dependency register AKR:**
A dependency register AKR for two histories EHA and EHB and a common history GH is a 1-tuple

AKR ($EH_A$, $EH_B$, GH) $\underline{def}$ (dependencyids)

- dependencyids are the identifiers for all recognized dependencies of the histories $EH_A$ and $EH_B$ and the common history GH.

All restrictions on decision options by existing conflicts of the first harmless conflict HK1 and of the second harmless conflict HK2 are recognized and marked at the start of conflict resolution on the basis of the
5  dependencies AK.

This is done using the decision restrictions' parameter introduced in the conflict definition for each conflict. All the possible restrictions are described
10  below.

The first harmless conflict HK1 impairs the decision options for conflicts with operations within the dedicated history. These dependent conflicts include
15  all those containing a createRelationship operation or a property change with the created data element. In this context, in the case of harmless conflicts HK6, ..., HK11 with property operations for this data element, the decisions are minimized by an adoption and
20  a creation of a new state.

The decision options for the critical conflicts KK1, KK2 and KK3 with relationship operations with the created data element are reduced by the option of
25  adopting the operations (-E3a, -E3b) for the copy database whose history contains the createElement operation. The decision options for the third harmless conflict HK3 and the fifth harmless conflict HK5 with relationship operations with the created data element
30  are reduced by the decision to accept and/or create a new state. Critical conflicts with property operations experience no modifications to their decision collection (decision sets).

35  A second harmless conflict HK2 impairs the decision option for conflicts in the two copy databases. The harmless conflicts for modifying properties HK6, ..., HK11 are treated as for the first harmless conflict HK1. The harmless conflicts with relationship

operations (HK4, HK5) in the history of the
deleteElement operation no longer have any decision
option for rejection and the decisions relating to the
harmless relationship conflicts in the other copy
5  database HK3, HK5 are minimized by the option of
accepting and/or creating a new state. The critical
conflicts with relationship operations in the copy
database without the deleteElement operation are
reduced by the option resetting and/or adopting the
10 operations. Critical conflicts with property operations
experience no modification to their property collection
in this case either.

A common deleteElement operation contained in the
15 common history GH and representing no conflict reduces
the decision options in specific cases. Thus, all
critical conflicts KK2 and KK3 with relationship
operations in which the target data element of the
common deleteRelationship operation corresponds to the
20 data element deleted on both sides can no longer be
reset.

Dependent conflicts with one or more relationship
operations (HK3, HK4, HK5, KK1, KK2, KK3), i.e. with a
25 plurality of listed identifiers and hence a plurality
of data elements involved, can have a plurality of
dependencies at the same time. In this regard, there
may be just one dependency for each arising identifier
in a conflict. A dependent conflict having a plurality
30 of identifiers can firstly have a plurality of
dependencies on conflicts of the same conflict type
(for example on two conflicts of the first harmless
conflict HK1a) and can secondly have a plurality of
dependencies on conflicts of different conflict types
35 (e.g. on a conflict of the type HK1a and on a conflict
of the type HK1b). By way of example, a dependent
conflict of the type HK3a can have two dependencies on
two conflicts of the type HK1a, namely one with the
identifier id1 and one with the identifier id2. At the

same time, the second critical contact KK2 can have a
dependency on the second harmless conflict HK2a, on the
first harmless conflict HK1a and on the first harmless
conflict HK1B.

5

The dependency of the identifiers for a conflict of the
type HK1 or HK2 means that there can be a maximum of
three restrictions per conflict of the type HK5, KK1,
KK2 or KK3. All conflicts of the type HK3A, HK3B, HK4A,
10    HK4B, HK5A and HK5B can have a plurality of
dependencies on conflicts of the same or different type
at the same time. Conflicts of the types KK1, KK2, KK3A
and KK3B, on the other hand, can have a plurality of
dependencies on conflicts of different types at the
15    same time. However, the same restrictions of the
decision options may also arise a number of times for
conflicts which have a number of dependencies on the
same conflict type. Thus, for a conflict of the type
HK3A, two dependencies are possible at the same time in
20    the presence of a conflict of the type HK1A or of one
of the type HK2B and the decisions are each minimized
by the first decision E1.

The multiple restrictions arising for many dependent
25    conflicts with the same decision options do not require
any separate consideration. Each of these decision
restrictions is considered as if it were an individual,
specific limitation. Hence, all multiple restrictions,
like others as well, are noted in the conflict. If a
30    conflict of the type HK1 and HK2 is resolved such that
one of the restrictions arising a number of times is
eliminated, the rest of these restrictions are
retained. Only when different conflict resolutions mean
that there are no longer any of the multiple
35    restrictions present can a decision of this type be
made. This applies irrespective of whether the
restriction was present once or a plurality of times.

The restrictions, recognized once at the start of reintegration, as a result of dependencies of the conflicts on conflicts of the type HK1 and HK2 are changed dynamically depending on the resolution of the conflicts of the type HK1 and HK2 during the reintegration. Thus, depending on the type of the dependent conflicts and on the respective resolution decision for the createElement operation and deleteElement operation, the following changes to the decision restrictions result:

a) The adoption of a createElement operation (first decision E1 relating to a conflict of the type HK1) causes the decision restrictions to be reset, i.e. the decision options to be extended, for all conflicts dependent on the operation.

b) The rejection of a deleteElement operation (second decision E2 relating to a conflict of the type HK2) likewise causes the decision restrictions to be reset, i.e. the decision options to be extended for the conflicts dependent on this conflict.

c) The rejection of a createElement operation (second decision E2 relating to a conflict of the type HK1) causes the decision restrictions already made for the conflicts dependent on this conflict to be retained. Hence, no decision options change.

d) The adoption of a deleteElement operation (first decision E1 relating to a conflict of the type HK2) causes the decision restrictions already made for the conflicts dependent on this conflict to be retained. Hence, no decision options change.

An anomaly created on one side can be recognized by means of two respective conflicts present in a history. In this context, there are the options of the conflict pairs HK5Aa/HK5Ab, HK4A/HK3A, HK4A/HK5A. In the case of a conflict of the type HK5Aa, the deleteRelationship operation is involved in the anomaly and in the case of a conflict of the type HK5Ab, the createRelationship operation is involved in the anomaly.

For anomalies arising on account of modifications on
the copy database of the second user B, the same
options apply: HK5Ba/HK5Bb, HK4B/HK3B, HK4B/HK5B. In
5    this context, a common feature of all conflict pairs is
that there is a deleteRelationship operation with an
identifier id1 as source data element and with an
identifier id2 as target data element in one of the two
conflicts and the same identifiers arise in reverse as
10   source data element and target data element in a
createRelationship operation for the other conflict.

The adoption of the conflict of the one-sided anomaly,
as described above, with the createRelationship
15   operation prevents rejection of the conflict of the
one-sided anomaly with the deleteRelationship operation
for a rejection of the conflict of the one-sided
anomaly with the deleteRelationship operation, but on
the other hand prevents adoption of the conflict of the
20   anomaly with the createRelationship operation. The
change in one of the two conflicts does not reduce the
decision options for the other conflict.

The result of this is that an anomaly created on one
25   side in directed relationships can be resolved by
rejecting the two conflicts, adopting the two
conflicts, changing the two conflicts differently or
modifying one of the conflicts and adopting or
rejecting the other conflict.
30

An anomaly created on both sides in directed
relationships can be resolved by deciding one of the
two conflicts and deciding the other conflict in a
different manner from this.
35

To prevent an anomaly (so-called pseudo-anomaly) from
arising, as described above, the following restrictions
on the decision options are provided:

a) After adoption of a conflict with the createRelationship operation containing the common data element id1 as target data element (createRelationship21), there must no longer be, for
5  the conflict with the two createRelationship operations containing the common data element as source data element, any decision option of the sixth decision option E6 with replacement of the target data element (idx or idz) by the source data
10  element id2(createRelationship21).

b) After a sixth decision E6 has been made on the basis of the sixth decision option E6 and selection of a new target data element id2 for the conflict with the two createRelationship operations containing the
15  common data element id1 as source data element, no further adoption of the conflict with the createRelationship operation with the common data element id1 as target data element (createRelationship21) must be possible.
20

As described above, as part of this method, the decision options for conflicts are limited on the basis of their dependencies, anomalies and pseudo-anomalies.

25  After each decision relating to a conflict, a change to the decision options for the conflicts which are dependent on the conflict just resolved or are situated in a common anomaly or pseudo-anomaly with this conflict is made on the basis of the dependencies,
30  anomalies and pseudo-anomalies.

A decision is made for each conflict, as described above. The decision can be made in different ways. An overview of possible decision variations can be found
35  in [1].

Within the scope of this illustrative embodiment, provision is made for a database or copy database to be regarded as a reference database, and for the

adjustment to be carried out on the basis of the reference database.

Thus, as shown in **Figure 1** by means of a recursive loop
5    via a checking step (step 105), a check is made to determine whether there is still any conflict and whether a decision thus needs to be made. If a decision is still to be made, then it is made. If there are no more conflicts present, then a last method step (step
10   106) is carried out, storage of the reintegrated database, which contains no more inconsistencies.

The inconsistency-free database is again transmitted to all further computers connected to the first computer
15   200 (step 107).

All the computers therefore have a consistent copy database.

20   The following text illustrates a few alternatives to the illustrative embodiment described above:

Inconsistencies can also be recognized after a prescribable number of elimination operations carried
25   out for an inconsistency by searching for a further inconsistency. This can be extended such that, each time an inconsistency has been eliminated, a subsequent inconsistency is again sought and eliminated.

30   It is also possible for the first computer to use the method illustrated above to ascertain a series of correction commands (correction sequences) which is respectively transmitted to the computer whose copy database has been checked for inconsistencies, and for
35   the respective computer to use the correction sequence to adjust its copy database to the database.

In addition, in one alternative embodiment, it is likewise possible to leave the decision to a user or to

a plurality of users, i.e. the decision options are
displayed to a user on the screen and the user uses the
keyboard or the computer mouse to select the decision
he requires, which is then implemented by the computer.

5

This document cites the following publication:

[1] DE 196 07 132 A1

**09/744225**

CLMS

J0o2 Rec'd PCT/PTO  2 2 JAN 2001

- 50 -

Patent claims

1.     A method for the computer-aided elimination of at least one inconsistency in a database collection containing a database and at least one copy database of the database, which inconsistency arises on account of the database and/or the copy database being changed,

a)  in which at least some of the operations which create an inconsistency are allocated to defined conflict types,

b)  in which each conflict type is allocated a decision set which is used to indicate possible decisions which can be used to eliminate an inconsistency created by at least one operation of the respective conflict type,

c)  in which the inconsistency is eliminated using the decision set.

2.     The method as claimed in claim 1, in which a plurality of inconsistencies are eliminated.

3.     The method as claimed in claim 1 or 2, in which each conflict type is allocated a decision set which is used to indicate possible decisions which can be used to eliminate an inconsistency created by a plurality of operations of the respective conflict type.

4.     The method as claimed in one of claims 1 to 3, in which the database collection contains a plurality of copy databases of the database.

5.     The method as claimed in one of claims 1 to 4, in which all the inconsistencies and their dependencies on one another are ascertained before the inconsistencies are eliminated.

6.     The method as claimed in one of claims 1 to 5, in which a conflict, an anomaly or a pseudo-anomaly is ascertained when an inconsistency is ascertained.

7.     The method as claimed in one of claims 1 to 6, in which, during elimination of the inconsistencies, the decision set for at least one conflict type is modified depending on the dependencies of the

inconsistencies.

8.      The method as claimed in one of claims 1 to 7, in which, after a prescribable number of eliminated inconsistencies, the database collection is examined for further inconsistencies and their dependencies, anomalies and pseudo-anomalies.

9.      The method as claimed in one of claims 1 to 8, in which the database collection contains an object-oriented database.

10.     The method as claimed in one of claims 1 to 9, used within the context of object-oriented software development.

11.     The method as claimed in one of claims 1 to 9, used within the context of creating a structured electronic document.

12.     An arrangement for eliminating at least one inconsistency in a database collection containing a database and at least one copy database of the database, which inconsistency arises on account of the database and/or the copy database being changed, having at least one processor which is set up such that the following steps can be carried out:

a)   at least some of the operations which create an inconsistency are allocated to defined conflict types,

b)   each conflict type is allocated a decision set which is used to indicate possible decisions which can be used to eliminate an inconsistency created by at least one operation of the respective conflict type,

c)   the inconsistency is eliminated using the decision set.

13.     The arrangement as claimed in claim 12, in which the processor is set up such that a plurality of inconsistencies are eliminated.

14.     The arrangement as claimed in claim 12 or 13, in which the processor is set up such that each conflict type is allocated a decision set which is used to indicate possible decisions which can be used to

- 52 -

eliminate an inconsistency created by a plurality of operations of the respective conflict type.

15. The arrangement as claimed in one of claims 12 to 14,

5 in which the processor is set up such that the database collection contains a plurality of copy databases of the database.

16. The arrangement as claimed in one of claims 12 or 15,

10 in which the processor is set up such that all the inconsistencies and their dependencies on one another can be ascertained before the inconsistencies are eliminated.

17. The arrangement as claimed in one of claims 12
15 to 16,

in which the processor is set up such that a conflict, an anomaly or a pseudo-anomaly can be ascertained when an inconsistency is ascertained.

18. The arrangement as claimed in one of claims 12
20 to 17,

in which the processor is set up such that, during elimination of the inconsistencies, the decision set for at least one conflict type can be modified depending on the dependencies of the inconsistencies.

25 19. The arrangement as claimed in one of claims 12 to 18,

in which the processor is set up such that, after a prescribable number of eliminated inconsistencies, the database collection is examined for further
30 inconsistencies and their dependencies, anomalies and pseudo-anomalies.

20. The arrangement as claimed in one of claims 12 to 19,

in which the processor is set up such that the database
35 collection contains an object-oriented database.

21. The arrangement as claimed in one of claims 12 to 20,

used within the context of object-oriented software development.

22.      The arrangement as claimed in one of claims 12
to 21,

used within the context of creating a structured
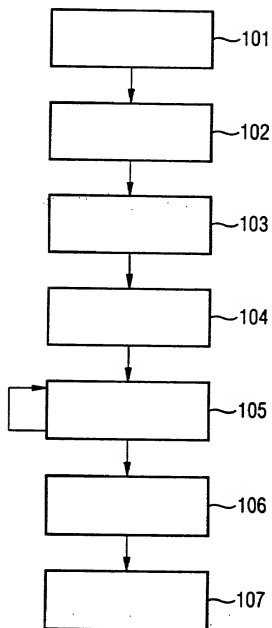electronic document.

5  23.      A set of a plurality of arrangements for
eliminating at least one inconsistency in a database
collection containing a database and at least one copy
database of the database, which inconsistency arises on
account of the database and/or the copy database being
10  changed,

in which each arrangement has at least one processor
which is set up such that the following steps can be
carried out:

a)  at least some of the operations which create an
15      inconsistency are allocated to defined conflict
       types,

b)  each conflict type is allocated a decision set
       which is used to indicate possible decisions which
       can be used to eliminate an inconsistency created
20      by at least one operation of the respective
       conflict type,

c)  the inconsistency is eliminated using the decision
       set, and

in which the arrangements can be coupled to one
25  another.

Abstract

**Method, arrangement and set of a plurality of arrangements for eliminating at least one inconsistency in a database collection containing a database and at least one copy database of the database**

To eliminate at least one inconsistency in a database collection containing a database and at least one copy database of the database, which inconsistency arises on account of the data in the database or in the copy database being changed, at least some of the operations which can create an inconsistency are allocated to defined conflict types. Each conflict type is allocated a decision set which is used to indicate possible decisions which can be used to eliminate an inconsistency created by an operation of the respective conflict type. The inconsistency is eliminated using the decision set. Error-free elimination of inconsistencies is ensured by matching the decision set to the respective situation.
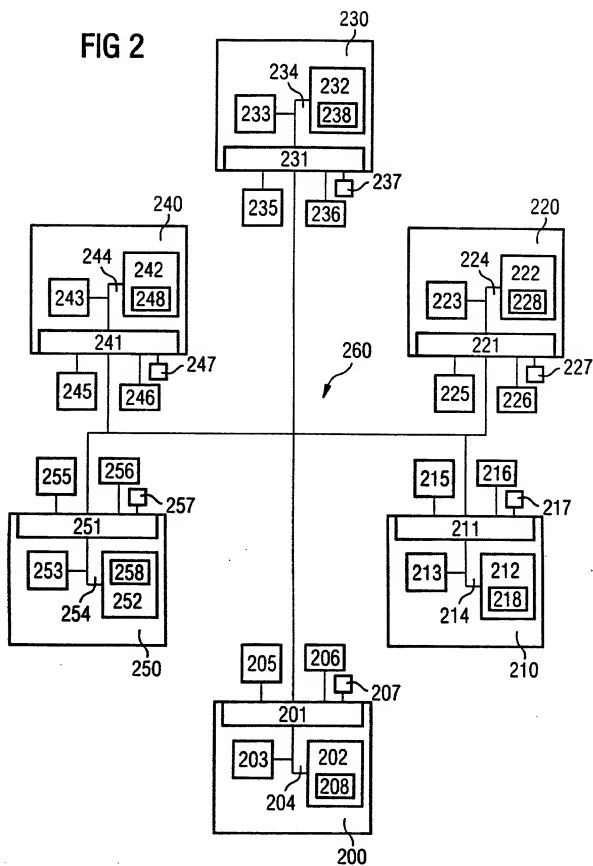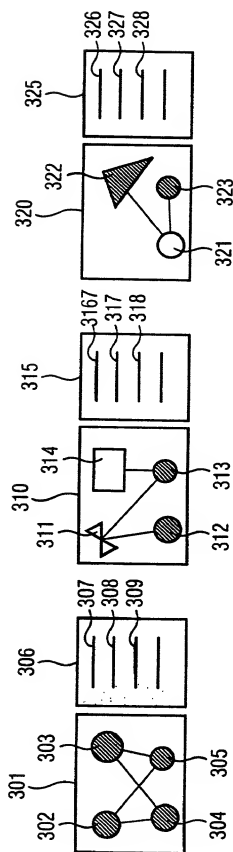
**FIG 1**

**FIG 2**

FIG 3

4/4

**FIG 4**

| | ES1 | ES2 | ES3 | ES4 | ES5 | ES6 | ES7 | ES8 | ES9 | ES10 | ES11 | ES12 | ES13 | ES14 | ES15 | ES16 | ES17 | ES18 | ES19 | ES20 | ES21 | ES22 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | H K1 | H K2 | H K3 | H K4 | H K5 | H K6 | H K 7I | H K 7II | H K 8I | H K 8II | H K9 | H K 10 | H K 11 | K K1 | K K2 | K K3 | K K4 | K K 5I | K K 5II | K K6 | K K7 | K K8 |
| E1 | × | × | × | × | × | × | × | × | × | × | × | × | × | | | | | | | | | |
| E2 | × | × | × | × | × | × | × | × | × | × | × | × | × | × | × | × | × | × | × | × | × | × |
| E3a | | | | | | | | | | | | | | × | × | × | × | × | × | × | × | × |
| E3b | | | | | | | | | | | | | | × | × | × | × | × | × | × | × | × |
| E4 | | | | | | | | ⊗ | | × | | | | | | | | | | | | |
| E5a | | | | | | | | | | | | | | | | | | | × | | | |
| E5b | | | | | | | | | | | | | | | | | | | × | | | |
| E6 | | | × | | × | × | × | × | × | × | × | | × | × | × | × | × | × | × | × | × | × |

SPECIFICATION

TITLE

METHOD, ARRANGEMENT AND SET OF A PLURALITY OF ARRANGEMENTS
FOR ELIMINATING AT LEAST ONE INCONSISTENCY IN A DATABASE
5      COLLECTION CONTAINING A DATABASE AND AT LEAST ONE COPY
DATABASE OF THE DATABASE


BACKGROUND OF THE INVENTION

Field of the Invention

10      The invention relates to a method, an arrangement and a set of a plurality of
arrangements for eliminating at least one inconsistency in a database collection
containing a database and at least one copy database of the database.


Description of the Related Art

15      Such a method is disclosed in German patent document DE 196 07 132 A1
('132) in which computers communicate with one another via a communication
network (e.g., a data network, a radio network or else a conventional telephone
network) using a communication protocol. A communication protocol means a
protocol for stipulating the data format used for communication between computers.
20  An exemplary communication protocol is the Transport Control Protocol/Internet
Protocol (TCP/IP).

In the method in '132, a first computer stores a database and each further
computer stores a copy of the database, called copy database below. The database
and the copy databases are modified by a respective computer during a session,
25  i.e., the data contained in the database or in a copy database, or the structure of this
data, is modified. In this context, a database means a hierarchical or else object-
oriented database, for example. A database contains data which is stored in
accordance with a prescribed structure and is interrelated. Each object, i.e., each
data record within the database, is usually unambiguously identifiable by means of
30  an identifier.


-1-          SUBSTITUTE SPECIFICATION

Changes to a copy database are sometimes made without the same change also being made in the database itself, or else vice versa. If a consistent database is intended to be created from the respective copy database and the database, then it is necessary to ascertain and eliminate any inconsistency arising as a result of the data, or the structure of said data, being added, removed or changed. An inconsistency means any syntactical difference within a copy database and the database, i.e., any discrepancies arising in the copy databases and the database between the data elements contained in the database and a copy database, the properties of these data elements and their relationships with one another. '132 indicates various options for eliminating such an inconsistency.

## SUMMARY OF THE INVENTION

The invention addresses the problem by specifying another method and another apparatus for eliminating inconsistencies in a database collection which allows an inconsistency to be eliminated while saving as much computation time as possible.

The problem is solved by the method, by the arrangement, and by the set of a plurality of arrangements configured to implement the following steps: changing the database or the at least one copy database, thereby producing an inconsistency; allocating at least some operations which create an inconsistency to defined conflict types; allocating each conflict type a decision set which is used to indicate possible decisions which can be used to eliminate an inconsistency created by at least one operation of the respective conflict type; and eliminating the inconsistency utilizing the decision set.

The method for the computer-aided elimination of at least one inconsistency in a database collection containing a database and at least one copy database of the database, which inconsistency arises due to a change in the database and/or in the copy database, has the following steps:

a)     at least some of the operations which can create an inconsistency are
       allocated to defined conflict types,

-2-          SUBSTITUTE SPECIFICATION

b) each conflict type is allocated a decision set which is used to indicate possible decisions which can be used to eliminate an inconsistency created by at least one operation of the respective conflict type, and

c) the inconsistency is eliminated using the decision set.

The arrangement for eliminating at least one inconsistency in a database collection containing a database and at least one copy database of the database, which inconsistency arises due to the data in the database or in the copy database being changed, has at least one processor which is set up such that the following steps can be carried out:

a) at least some of the operations which can create an inconsistency are allocated to defined conflict types,

b) each conflict type is allocated a decision set which is used to indicate possible decisions which can be used to eliminate an inconsistency created by an operation of the respective conflict type, and

c) the inconsistency is eliminated using the decision set.

The set of a plurality of arrangements for eliminating at least one inconsistency in a database collection containing a database and at least one copy database of the database, which inconsistency arises due to the database and/or the copy database being changed, contains a plurality of arrangements, each of which has at least one processor which is set up such that the following steps can be carried out:

a) at least some of the operations which can create an inconsistency are allocated to defined conflict types,

b) each conflict type is allocated a decision set which is used to indicate possible decisions which can be used to eliminate an inconsistency created by at least one operation of the respective conflict type, and

c) the inconsistency is eliminated using the decision set.

The arrangements can be coupled to one another.

The invention makes it possible to generically resolve an inconsistency in a complex database. Preferred developments of the invention are described below.

In one preferred embodiment, a plurality of inconsistencies are eliminated. Preferably, in a further embodiment, each conflict type is allocated a decision set which is used to indicate possible decisions which can be used to eliminate an inconsistency created by a plurality of operations of the respective conflict type. In addition, in one development, the database collection contains a plurality of copy databases of the database.

For the purposes of simplification and hence to save computation time when eliminating an inconsistency, one embodiment of the invention provides for all inconsistencies and their dependencies on one another to be ascertained before elimination. A further saving of the computation time required to eliminate a plurality of inconsistencies is achieved in a further embodiment by virtue of the decision set for at least one conflict type being modified during elimination of the inconsistencies. In this context, the respective decision set is preferably changed on the basis of dependencies of the inconsistencies.

In one preferred embodiment, after a prescribable number of eliminated inconsistencies, the database collection is examined for further inconsistencies and their dependency. In one embodiment, the database collection preferably contains an object-oriented database. Finally, the method may be used within the context of object-oriented software development or else in the context of creating a structured electronic document.

## BRIEF DESCRIPTION OF THE DRAWINGS

An illustrative embodiment of the invention is shown in the figures and is explained in more detail below.

Figure 1     is a flowchart showing the method steps of the illustrative embodiment;

Figure 2     is a block diagram showing computers connected to one another via a communication network;

Figure 3     is a pictorial diagram of a database structure;

Figure 4     is a table presenting an overview of possible conflicts and decision sets with decision options for eliminating the respective conflicts.

–4–     SUBSTITUTE SPECIFICATION

## DESCRIPTION OF THE PREFERRED EMBODIMENTS

Figure 2 shows a first computer 200 having a memory 202 and a processor 203 which are connected to one another and to an input/output interface 201 via a bus 204 in each case. The input/output interface 201 is used to connect the first computer 200 to a screen 205, to a keyboard 206, and to a computer mouse 207. In addition, the first computer 200 is connected to further computers 210, 220, 230, 240 and 250 via a communication network 260, in the example, an ISDN network (Integrated Services Digital Network).

The first computer 200 stores a database 208. The further computers 210, 220, 230, 240 and 250 each likewise have a processor 213, 223, 233, 243 and 253 and each have a memory 212, 222, 232, 242 and 252. The processor 213, 223, 233, 243 and 253 and the memory 212, 222, 232, 242 and 252 are each connected via a respective bus 214, 224, 234, 244 and 254 to the communication network 260 via an input/output interface 211, 221, 231, 241 and 251. In addition, the further computers 210, 220, 230, 240 and 250 are each connected to a screen 215, 225, 235, 245 and 255 and to a keyboard 216, 226, 236, 246 and 256 and to a computer mouse 217, 227, 237, 247 and 257.

A respective copy of the database 208, called a copy database 218, 228, 238, 248 and 258 below, is transmitted from the first computer 200 to a respective further computer 210, 220, 230, 240 and 250 and is stored there in the respective memories 212, 222, 232, 242 and 252. Once the copy databases 218, 228, 238, 248 and 258 have been transmitted, the computers 200, 210, 220, 230, 240 and 250 interrupt communication and an independent change is made among the respective computers 200, 210, 220, 230, 240 and 250, i.e., the removal or addition of data ,or the removal or addition of dependencies of the data in a copy database 218, 228, 238, 248 and 258 and in the database 208.

Once communication between the first computer 200 and the further computers 210, 220, 230, 240 and 250 is resumed, a consistent database must be formed from the database 208 and the copy databases 218, 228, 238, 248 and 258. For this purpose, it is necessary to establish respective changes made in the

–5–     SUBSTITUTE SPECIFICATION

database 208 or in the copy databases in order to ascertain inconsistencies between the copy databases and the database 208 so that the inconsistencies can be eliminated. Independently of the syntactical structure and of the dependencies of the data elements on one another, each data element can have an arbitrary number of properties. In this context, each property is of a particular property type and is represented by a current value. For all properties, it is assumed with regard to the value ranges that all values may comprise only symbols or compiled symbols from an ASCII table (digits, numbers, letters, special characters, character chains). A series of such characters and symbols is called an entry below. More complex properties are represented in the application modeling by data elements and relationships.

Three types of properties for the syntactical analysis are distinguished below on the basis of the operations which can be carried out on the property:

- **Individual value:**

  A "value" as a property type describes an individual entry, the entry always being regarded in its entirety and also being modified as such. In this context, the "value" type property is always modified by complete replacement of the entry for the property with a new entry.

- **Listing:**

  A "listing" as a property type describes a quantity of arbitrary entries, the entries bearing no relation to one another and, for their part, being able to represent an individual value, a listing, or an ordered listing. In this context, the individual entries can be added or deleted only individually. The unambiguity of the entries must be ensured if requested by the application. An example of a data structure representing this property type is a hash table or an array.

- **Ordered listing:**

  Properties of the "ordered listing" type describe a quantity of arbitrary entries, like properties of the simple listing type. In this case, however, the entries are in a defined order with respect to one another, this order being stipulated by way of an index for each entry. The indices are stipulated relative to the start of the listing. An insertion operation with individual entries or with a plurality of entries therefore

SUBSTITUTE SPECIFICATION

always relates to an index. A deletion operation can relate to an individual entry with only one index or to a series of successive entries, and hence to a start index and an end index. The criterion for the order is defined by an application program, and the observance of the order criteria is also monitored by the latter. An example of this property type is an indexed list containing arbitrary entries (e.g. text document), in which each line or each character corresponds to one entry.

A data element DE means a 4-tuple defined as follows:

**Data element**

A data element DE is a 4-tuple

DE *def* (ID, infospace, elementtype, properties);

- ID is a one-to-one identifier throughout the system
- infospace $\in$ MIR; where MIR is a quantity of all the information spaces
- elementtype $\in$ ET; where ET is a quantity of all the data element types
- properties $\subseteq$ {(name, propertytype, value):
- name $\in$ MEN, value $\in$ MEW, propertytype $\in$ MET},

  where:

  MEN is a quantity of all the property names and the following applies:

  $\forall\, i \in \{1,...,n\};\ \forall\, k \in \{1,...,m\}: name_i \neq name_k,$

  MEW is a quantity of all the property values, and

  MET is a quantity of all the property types

  {"value", "listing", "ordered listing"}.


In addition, an information space is defined as follows:

**Information space**

An information space IR is a 3-tuple

IR *def* (ID, irname, owner, data)


- ID is a one-to-one identifier throughout the system, where the following applies:

  $\forall\, i \in (1,...,n);\ \forall\, k \in (1,...,n): i \neq k:$

  IRi.ID $\neq$ IRk.ID;

  where MIR is a quantity of all the IRs present in the system and n is their number;

SUBSTITUTE SPECIFICATION

- irname $\in$ MIRN

  where the following applies:

  $\forall i \in (1, ..., m); \forall k \in (1, ..., m): i \neq k:$

  Iri.irname $\neq$ Irk.irname;

  where MIR is the quantity of all the IRs present in the system and m is their number;

  MIRN represents a quantity of all the possible information space names;

- owner = Ni where Ni $\in$ MN or Ngi where Ngi $\in$ MNG;

- data is the data which can be accessed by a user group and is associated with the IR.

A relationship between the data elements is additionally defined as follows:

**Relationship between data elements**

A relationship BZ between data elements is a 3-tuple

BZ *def* (relationshiptype, name, dataelement1, dataelement2)

- name $\in$ MBN; where MBN is a quantity of all the relationship names

- relationshiptype $\in$ MBT; where MBT is a quantity of all the relationship types {"undirected", "logical", "successor", "subsup"}

- dataelement1, 2 $\in$ MDE; where MDE is a quantity of all the data elements.

In order to ascertain inconsistencies, each computer 200, 210, 220, 230, 240 and 250 carries a respective protocol relating to all the operations carried out on the database or on the respective copy database and stores it in the form of a list. The stored list is called the history below. Hence, the database 208 and each copy database 218, 228, 238, 248 and 258 have a respective associated history.

This situation is shown in Figure 3, which shows the database 301 containing objects 302, 303, 304 and 305 and also a history 306 storing, as entries 307, 308, 309, change operations which have been carried out on the database 301 by the first computer 200 since communication with the further computers 210, 220, 230, 240 and 250 was interrupted. The entries 307, 308, 309 are likewise stored in the memory 202 of the first computer 200.

SUBSTITUTE SPECIFICATION

A first copy database 310 containing objects 311, 312, 313 and 314 likewise has an associated history 315 with appropriate change operations 316, 317, 318. The copy database 310 is stored in the further computer 210. A second copy database 320 containing objects 321, 322, 323 and its associated history 325 with change operations 326, 327, 328 is stored in a further computer 220. To form a consistent database, i.e., to reintegrate all the copy databases 218, 228, 238, 248 and 258 with the database 208, the histories 315, 325, etc. are transmitted to the first computer 200 via the communication network 260 and are stored in the memory 202 of the first computer 200. At the start of reintegration, which is described in Figure 1 by step 101, all the histories of the copy databases are transmitted to the first computer and are stored there (step 102). In a third step (step 103), all the histories 315, 325, etc. to be taken into account for reintegration are determined.

In addition, the following change operations are taken into account and are used to uniquely describe the inconsistencies which have arisen. Within the context of this illustrative embodiment, the following nine operations are taken into account as change operations, which are described below in the form of a pseudo-program code:

**1. Create Element:**

**Create Element** (R(IR), ID, Element type) → R(IR)

**createElement**(ir, id, elementtype) RETURN R(IR)

```
    BEGIN element := instantiate(elementtype)
        element.elementname:= id
        R(ir):= insert(R(ir).data,element)
        R(ir):= add(R(ir).history,
        "createElement(id,elementtype)")
        return R(ir)
    END
```

This operation creates a data element of the data type **elementtype** with the identifier **id** in a copy database **R(ir)** within an information space **ir**, to which this operation is applied. In this context, all of the properties of the newly created data element receive a preset initialization value. Having been initialized, the new

SUBSTITUTE SPECIFICATION

element is added under the specified name to the data in the copy database **R(ir)**
and the executed operation is added without the information space as a parameter
into the history **R(ir).history** associated with the copy database. In this context, the
unique identifier **id** can be created by the application sending the operation.

5

**2. DeleteElement:**

**DeleteElement(R(IR),ID) → R(IR)**

**deleteElement(ir,id)RETURN R(IR)**

     BEGIN element:= select (R(ir).data,id)

10        R(ir):= remove(R(ir).data, element)

       R(ir):= add(R(ir).history,"deleteElement(id)")

       return R(ir)

     END

       This operation deletes a data element with the name **id** from the copy
15 database **R(ir)** of the information space **ir** and writes the executed operation into the
history **R(ir).history** associated with the copy database. All the properties of the
data element which have been modified since the instantiation of the data element
are also lost in the process. The relationships affecting the element are retained,
however. If these are likewise to be deleted, then the application program is
20 responsible for this.

**3. ChangeProperty:**

**ChangeProperty(R(IR), ID, Property type, Value) → R(IR)**

**changeProperty(ir, id, propname, newValue) RETURN R(IR)**

25      BEGIN element := select (R(ir).data,id)

       property := select(element.properties,

       propname)

       property.value := newValue

       R(ir):= add (R(ir).history, "changeProperty

30        (id, propname, newValue)")

       return R(ir)

     SUBSTITUTE SPECIFICATION

END

This operation sets the value of the property **propname** of the data element with the identifier **id** to the value **newValue** in the copy database **R(ir)** of the information space **ir** and writes the executed operation into the history **R(ir).history** associated with the copy database.

**4. ChangePropertyAdd:**

**ChangePropertyAdd**(R(IR),ID,Property type,Entry)→R(IR)

**changePropertyAdd**(ir,id,propname,newEntry) RETURN R(IR)

BEGIN element := select (R(ir).data, id)

property := select (element.properties,

propname)

add (property.value, newEntry)

R(ir) := add(R(ir).history,

"changePropertyAdd (id, propname,

newEntry)")

return R(ir)

END

This operation is for properties of the "listing" type. The operation adds a new entry **newEntry** to the copy database **R(ir)** of the information space **ir** in the value of the property **propname** of the data element with the identifier **id** at the end of the listing. The executed operation is then stored in the history **R(ir).history** associated with the copy database.

**5. ChangePropertyDel:**

**ChangePropertyDel**(R(IR), ID, Property type, Index,

Entry) → R(IR)

**changePropertyDel**(ir, id, propname, index, oldEntry)

RETURN R(IR)

BEGIN element := select (R(ir).data, id)

property := select (element.properties,

-11-      SUBSTITUTE SPECIFICATION

```
                propname)
                del (property.value, oldEntry)
                R(ir) := add(R(ir).history,
                "changePropertyDel(id, propname, index,
5               oldEntry)")
                return R(ir)
        END
```

This operation is for properties of the "listing" type. The operation deletes the first entry **oldEntry** occurring in the listing in the copy database **R(ir)** of the information space **ir** in the value of the property **propname** of the data element with the identifier **id**. The executed operation is then stored in the history **R(ir).history** associated with the copy database.

**6. ChangePropertyInsert:**

```
15  ChangePropertyInsert(R(IR), ID, Property type, Index,
                Number, Entries) → R(IR)
    changePropertyInsert(ir, id, propname, index, number,
                newEntries) RETURN R(IR)
        BEGIN element := select (R(ir).data, id)
20              property := select (element.properties,
                propname)
                for (i=0, i < number, i++){
                incrIndex (property.value.entries,ind >=
                index)
25              insert (property.value.index, newEntries.
                (number -i))}
                R(ir) := add (R(ir).history,
                "changePropertyInsert (id, propname, index,
                number, newEntries)")
30              return R(ir)
        END
```

A property to which this operation can be applied is of the "ordered listing" type. This operation inserts a number number of new entries newEntries.i in the copy database **R(ir)** of the information space **ir** in the value of the property **propname** of the data element with the identifier **id** from the position Index in the ordered listing Property.Value. For all the entries in the ordered listing **property.Value** having an identical index to or a larger index than **index, index** is increased by the value **number**. The executed operation is then stored in the history **R(ir).history** associated with the respective copy database.

**7. ChangePropertyRemove:**

**ChangePropertyRemove**(R(IR), ID, Property type, Index,
              Number, Entries) → R(IR)
**changePropertyRemove**(ir, id, propname, index, number,
              oldEntries) RETURN R(IR)
    BEGIN element := select (R(ir).data,id)
        property := select (element.properties,
        propname)
        for (i=0, i < number, i++){
        remove (property.value.index, oldEntries.
        (i+1))
        decrIndex (property.value.entries, ind >
        index)}
        R(ir) := add(R(ir).history,
        "changePropertyRemove(id, propname, index,
        number, oldEntries)")
        return R(ir)
    END

A property to which this operation can be applied is of the "ordered listing" type. This operation deletes the entries **oldEntries** in the copy database **R(ir)** of the information space **ir** in the value of the property **propname** of the data element with the identifier **id** from the position Index in the ordered listing **Property.Value**. All

entries with a larger index than **(index + number)** have their index reduced by the value **number**. The executed operation is then stored in the history **R(ir).history** associated with the respective copy database.

**8. CreateRelationship:**

**CreateRelationship**(R(IR), Name, Relationship type,

ID1, ID2)→R(IR)

**createRelationship**(ir, name, reltype, fromid, toid,

toidir) RETURN R(IR)

BEGIN relationship := instantiate (reltype)

relationship.name := name

relationship.dataelement1 := fromid

relationship.dataelement2 := toid

relationship.dataelement2.ir := toidir

R(ir) := insert (R(ir).data, relationship)

R(ir) := add (R(ir).history,

"createRelationship (name, reltype, fromid,

toid, toidir)")

return R(ir)

END

This operation creates a relationship of the **reltype** type between the data elements with the identifiers **fromid** and **toid** under the name **name** and adds the new relationship to the data in the copy database **R(ir)** of the information space **ir**. The executed operation is then stored in the history **R(ir).history** associated with the copy database. It is assumed for all relationships that, for each relationship name allocated, there is only a single relationship of the same type between two data elements. If a plurality of relationships of the same type are necessary under the same name between two data elements, then identifiers need to be introduced for relationships as well. However, the assumption made is sufficient for the majority of applications. The information space of the target data element need be specified only in the case of a logically externally directed relationship.

SUBSTITUTE SPECIFICATION

**9. DeleteRelationship:**

**DeleteRelationship**(R(IR), Name, Relationship type, ID1,

        ID2)→R(IR)

5    **deleteRelationship**(ir, name, reltype, fromid, toid.toidir)

        RETURN R(IR)

     BEGIN relationship := select (R(ir).data, name,

        reltype, fromid, toid)

       R(ir) := remove (R(ir).data, relationship)

10       R(ir) := add (R(ir).history,

       "deleteRelationship (name, reltype, fromid,

       toid.toidir)")

       return R(ir)

    END

15       This operation deletes a relationship of the **reltype** type between the data elements with the identifiers **fromid** and **toid** under the name **name** from the copy database **R(ir)** of the information space **ir**. The executed operation is then stored in the history **R(ir).history** associated with the copy database. The information space of the target data element need be specified only for the relationships of the logically

20  externally directed type.

      In a further step, all conflicts, dependencies, anomalies, pseudo-anomalies and restrictions by dependencies are recognized (step 103). A conflict means the smallest decidable quantity of operations which arise only on one side in syntactical terms, uniquely describe an inconsistency and can be meaningfully presented to a

25  user or to the system and eliminated (decided) by the latter. Each conflict can be recognized as a whole and can be resolved by a single decision during reintegration. Potential conflicts are subsequently defined on the basis of the data structure and the operations occurring in the history.

      A distinction is drawn between harmless conflicts and critical conflicts.

30  Harmless conflicts (HK) contain only operations which describe modifications on one copy database. In this case, there is therefore just one user desiring and making a

   SUBSTITUTE SPECIFICATION

change to the part of the data structure or the copy database or else the database itself. The operations carried out are thus complemented. Depending on which copy database is to be allocated the operation or operations, when the copy databases of users A and B are present, harmless conflicts with operations on the copy database

5   of a first user A can be referred to as HKA and harmless conflicts with operations on the copy database of a second user B can be referred to as HKB, on a general basis.

By contrast, critical conflicts (KK) contain changes on both sides to the same part of the data structure and represent contrary views of the users about the

10  ultimate state of particular data within the database and copy databases. In this context, a critical conflict can also be defined by different operations on two copy databases. In such cases, a distinction is drawn between a critical conflict KKA and a critical conflict KKB.

Formally, a conflict is defined as follows:

15  **Conflict:**

A conflict K between two histories EHA and EHB and a common history GH is a 6-tuple

K(EHA, EHB, GH) *def*

(id, ktype, operationsEHA, operationsEHB,

20      operationsGH, decisionrestr);

- id is a one-to-one identifier throughout the system (see also the definition of a data element)
- ktype ∈ {HK1A, ..., HK11A, HK1B, ..., HK11B, KK1, KK2, KK3A, KK3B, KK4, KK5A, KK5B, KK6, KK7, KK8A,

25      KK8B}

- operationsEHA ∈ EHA.operations;
- operationsEHB ∈ EHB.operations;
- operationsGH ∈ GH.operations;
- decisionrestr ⊆ MENTktype; where MENTktype is a quantity of all the possible

30      decisions for a conflict of the type ktype.

The conflicts are shown in Figure 4.

**1. First harmless conflict HK1:**

HK1 = (createElement/ -- )

        There is a creation operation for a data element createElement(id, elementtype) in one history only.

HK1A = createElement(id, elementtype) $\in$ EHA $\vee$

HK1B = createElement(id, elementtype) $\in$ EHB.

**2. Second harmless conflict HK2:**

HK2 = (deleteElement/ -- )

        There is a deletion operation for a data element deleteelement(id, elementtype) in one history only.

HK2A = deleteElement(id, elementtype) $\in$ EHA $\vee$

HK2B = deleteElement(id, elementtype) $\in$ EHB.

**3. Third harmless conflict HK3:**

HK3 = (createRelationship/ -- )

        There is a creation operation for a relationship createRelationship (reltype, rname, id1, id2) in one history only.

HK3A = createRelationship(reltype, rname, id1, id2) $\in$ EHA $\vee$

HK3B = createRelationship(reltype, rname, id1, id2) $\in$ EHB.

**4. Fourth harmless conflict HK4:**

HK4 = (deleteRelationship/ -- )

        There is a deletion operation for a relationship deleteRelationship (reltype, rname, id1, id2) in one history only.

HK4A = deleteRelationship(reltype,rname,id1,id2) $\in$ EHA $\vee$

HK4B = deleteRelationship(reltype,rname,id1,id2) $\in$ EHB.

**5. Fifth harmless conflict HK5:**

HK5 = (deleteRelationship12, createRelationship 13/ -- )

    SUBSTITUTE SPECIFICATION

There is a deletion operation for a relationship deleteRelationship(reltype, rname id1, id2) and a subsequent creation operation for the relationship creationRelationship(reltype, rname, id1, id3) from the same source data element to another target data element in one history only.

5  HK5A = [deleteRelationship(reltype,rname,id1,id2),
       createRelationship(reltype,rname,id1,id3)] $\in$ EHA $\vee$
   HK5B = [deleteRelationship(reltype,rname,id1,id2),
       createRelationship(reltype,rname,id1,id3)] $\in$ EHB.


10 **6. Sixth harmless conflict HK6:**

   HK6 = (changeProperty/ -- )

   There is a change operation changeProperty(id, name, valuenew, valueold) for a property of the "Value" type in one history only.

   HK6A = changeProperty(id,name,value1,value0) $\in$ EHA $\vee$
15 HK6B = changeProperty(id,name,value1,value0) $\in$ EHB.


   **7. Seventh harmless conflict HK7:**

   HK7 = (n x changePropertyAdd/ -- )

   There are n (n is an element of the natural numbers and n > 0) insertion
20 operations changePropertyAdd(id, name, entry) with the same entry **entry** for the same property of the "listing" type for a data element in one history, with there being no deletion operation with the same entry for the same property of the data element in another history. The inconsistency described consists in the fact that there are n entries of the type **entry** more in the copy database with the creation operations in
25 the history associated with the copy database than in the other copy database.

   HK7A = n times changePropertyAdd(id,name,entry) $\in$ EHA $\vee$
   HK7B = n times changePropertyAdd(id,name,entry) $\in$ EHB.


   **8. Eighth harmless conflict HK8:**
30 HK8 = (n x changePropertyDel/ -- )

There are n (n is an element of the natural numbers) deletion operations changePropertyDel(id, name, entry) with the same entry **entry** for a property of the "listing" type for a data element in one history, with there being no insertion operation with the same entry for the property of the data element in another history.

5 The inconsistency described consists of the fact that the n entries of the type **entry** are present to a lesser extent in the copy database whose history contains the deletion operations than in the other copy database.

HK8A = n times changePropertyDel(id,name,entry) ∈ EHA ∨
HK8B = n times changePropertyDel(id,name,entry) ∈ EHB.

10

## 9. Ninth harmless conflict HK9:

HK9 = (changePropertyInsert/ -- )

There is an insertion operation changePropertyInsert(id, name, index1, 1, entry1) for an index with an individual entry for a property of the "ordered listing" type

15 for a data element in one history only, with another history containing no insertion operations with another entry for a checkable identical index for the same property of the data element.

HK9A = changePropertyInsert(id,name,index,1,entry)

∈ EHA ∨

20 HK9B = changePropertyInsert(id,name,index,1,entry)

∈ EHB.

## 10. Tenth harmless conflict HK10:

HK10 = (changePropertyRemove/ -- )

25 There is a deletion operation changePropertyRemove(id, name, index1, 1, entry1) for an index with an individual entry for a property of the "ordered listing" type for a data element in one history only.

HK10A = changePropertyIRemove(id,name,index,1,entry)

∈ EHA ∨

30 HK10B = changePropertyIRemove(id,name,index,1,entry)

∈ EHB.

SUBSTITUTE SPECIFICATION

**11. Eleventh harmless conflict HK11:**

HK11 = (changePropertyRemove,changePropertyInsert/--)

There is a deletion operation changePropertyRemove(id, name, index1, 1, entry1) for an index **index** with an individual entry for a property of the "ordered listing" type for a data element, and a subsequent creation operation changePropertyInsert (id, name, index1, 1, entry2) for an entry for the same index for the same property of the same data element in one history only.

HK11A = [changePropertyIRemove(id,name,index,1,entry1),

changePropertyIInsert(id,name,index,1,entry2)]

∈ EHA ∨

HK11B = [changePropertyIRemove(id,name,index,1,entry1),

changePropertyIInsert(id,name,index,1,entry2)]

∈ EHB.

The following text gives an overview of critical conflicts (KK), i.e. operations in a plurality of histories:

**1. First critical conflict KK1:**

KK1 = (createRelationship12/ createRelationship13)

There is a creation for a relationship createRelationship(reltype, rname, id1, id2) in one history, with a creation createRelationship(reltype, rname, id1, id3) for the same relationship (reltype, rname), starting from the same source data element but to another target data element, existing in another history.

KK1 = createRelationship(reltype,rname,id1,id2) ∈ EHA ∧

createRelationship(reltype,rname,id1,id3) ∈ EHB.

**2. Second critical conflict KK2:**

KK2 = (deleteRelationship12, createRelationship13 /

deleteRelationship12, createRelationship14)

The different change in a relationship can be recognized in the histories like the first critical conflict KK1. In addition, however, a common history GH contains a deletion deleteRelationship(reltype, rname, id1, id2) for the common relationship (rname, reltype) from the same source data element, but to another target data element.

KK2 = deleteRelationship(reltype,rname,id1,id2) ∈ GH ∧
    createRelationship(reltype,rname,id1,id3) ∈ EHA ∧
    createRelationship(reltype,rname,id1,id4) ∈ EHB.

**3. Third critical conflict KK3:**

KK3 = (deleteRelationship12, createRelationship13/
    deleteRelationship12)

A relationship's modification on one side and deletion on the other side can be recognized in the histories, like the third harmless conflict HK3, by an operation createRelationship(reltype, rname, id1, id3). In addition, however, the common history contains a deletion

deleteRelationship (reltype, rname, id1, id2) for the common relationship (rname, reltype) from the same source data element but to the last common target data element.

KK3A = deleteRelationship(reltype,rname,id1,id2) ∈ GH ∧
    createRelationship(reltype,rname,id1,id3) ∈ EHA;
KK3B = deleteRelationship(reltype,rname,id1,id2) ∈ GH ∧
    createRelationship(reltype,rname,id1,id3) ∈ EHB.

**4. Fourth critical conflict KK4:**

KK4 = (changeProperty/ changeProperty)

There is a change operation changeProperty (id, name, valuenew1, valueold) for a property of the "value" type for a data element in one history, with another change operation existing in another history for the same property of the data element.

KK4 = changeProperty(id,name,valuenew1, valueold) ∈ EHA ∧

SUBSTITUTE SPECIFICATION

changeProperty(id,name,valuenew2, valueold) ∈ EHB.

### 5. Fifth critical conflict KK5:

KK5 = (n changePropertyAdd/ m changePropertyDel)

There are n (n is an element of the natural numbers) identical operations of the type

changePropertyAdd(id, name, entry) with the same entry for a property of the "listing" type for a data element in one history, with another history m (m is an element of the natural numbers) containing identical operations of the type changePropertyDel(id, name, entry) with the same entry for the same property of the data element. The inconsistency described consists of the fact that the copy database with the creation operations in the history associated with the copy database contains n + m identical entries entry more than the other copy database. To be able to make an exact statement about the occurrence of the operations, a distinction is drawn for the fifth critical conflict KK5 between a fifth critical conflict of first type KK5A and a fifth critical conflict of second type KK5B. In this case, the allocation is made using the creation operations.

KK5A = (n changePropertyAdd(id,name,entry) ∈ EHA ∧
    m changePropertyDel(id,name,entry) ∈ EHB ∨

KK5B = (m changePropertyDel(id,name,entry) ∈ EHA;
    n changePropertyAdd(id,name,entry) ∈ EHB.

### 6. Sixth critical conflict KK6:

KK6 = (changePropertyInsert / changePropertyInsert)

There is an insertion operation changePropertyInsert(id, name, index1, 1, entry1) for an index with an individual entry for a property of the "ordered listing" type for a data element in one history, with another history containing an insertion operation with another entry for the same (this can be checked) index for the property of the data element.

KK6 = changePropertyInsert(id,name,index1,1,entry1)
    ∈ EHA ∧

changePropertyInsert(id,name,index2,1,entry2)

    ∈ EHB.


### 7. Seventh critical conflict KK7:

KK7 = (changePropRemove, changePropInsert /

    changePropRemove, changePropInsert)

    The different change on the two sides in an individual entry on a common index **index** for a property of the "ordered listing" type for a data element can be recognized in the histories like the sixth critical conflict KK6. In addition, however, the common history contains a changePropertyRemove(id, name, index1, 1, entry1) operation for the last common entry on the same (this can be checked) index.

KK7 = changePropertyRemove(id,name,index1,1,entry1)

    ∈ GH ∧

changePropertyInsert(id,name,index2,1,entry2)

    ∈ EHA ∧

changePropertyInsert(id,name,index3,1,entry3)

    ∈ EHB.


### 8. Eighth critical conflict KK8:

KK8 = (changePropRemove, changePropInsert /

changePropRemove)

    The change, on one side, and deletion, on the other side, in/of an individual entry on a common index **index** for a property of the "ordered listing" type for a data element in the histories can be recognized like a tenth harmless conflict HK10. In addition, however, the common history contains a changePropertyRemove(id, name, index1, 1, entry1) operation for the last common entry on the same (this can be checked) index.

KK8A = changePropertyRemove(id,name,index1,1,entry1)

    ∈ GH ∧

changePropertyInsert(id,name,index2,1,entry2)

    ∈ EHA;

    SUBSTITUTE SPECIFICATION

KK8B = changePropertyRemove(id,name,index1,1,entry1)

  ∈ GH ∧

changePropertyInsert(id,name,index2,1,entry2)

  ∈ EHB.

5

      The operations stored in the histories describe the autonomously modified data areas directly and are used to describe and to eliminate (described below) the inconsistencies. To recognize the inconsistencies, two respective histories are compared with one another. The inconsistencies are recognized at the start of the

10  method, before the actual reintegration.

      The search for the existing inconsistencies in the copy databases by searching for conflict operations is carried out in the three steps described below.

- A first step makes a pass through the two histories to be compared with one another in the copy databases and in the database, which are to be matched to

15    one another. All of the operations in the histories are allocated, separately on each side, to a respective one of the new operation collections (createElement operations, deleteElement operations, createRelationship operations, deleteRelationship operations, changeProperty operations, changePropertyAdd operations, changePropertyDel operations, changePropertyInsert operations and

20    changePropertyRemove operations) described above.

- In a second step, a respective conflict register KR is started for each conflict type described above HK1A through HK11A, HK1B through HK11B, KK1, KK2, KK3A, KK3B, KK4, KK5A, KK5B, KK6, KK7, KK8A, KK8B. In this context, it is ensured that all conflicts for which operations from the two histories contribute to the

25    respective conflict are not recognized twice and stored in the respective conflict register KR twice. Accordingly, the operation collections just formed are searched through on the basis of the definitions of the conflict types, as described above, starting with the first harmless conflict HK1A in the history of the first user A. If a conflict has been ascertained, the conflict is stored in the conflict register KR for

30    the appropriate conflict type, for example, the first harmless conflict HK1 is stored in the copy database of the first user A in a conflict register KR_HK1A.

- If the search and storage of the conflicts has taken place, the operation collections created at the start are deleted again in a third step. The subsequent elimination of the inconsistencies is based on the conflicts stored in the conflict registers KA and on the operations of these conflicts.

A conflict register KR is defined as follows:

**Conflict register KR:**

A conflict register KR for two histories EHA and EHB and a common history GH is a 2-tuple

$KR(EH_A, EH_B, GH)$ *def* (crtype, conflictids)

- crtype $\in$ {KA_HK1A, ..., KA_HK11A, KA_HK1B, ..., KA_HK11B, KA_KK1, KA_KK2, KA_KK3A, KA_KK3B, KA_KK4, KA_KK5A, KA_KK5B, KA_KK6, KA_KK7, KA_KK8A, KA_KK8B}

- conflictids are identifiers for all the conflicts K(EHA, EHB, GH) associated with the conflict register KR, where K.type can be allocated to the respective conflict array type KR.crtype.

An anomaly is present when two data elements exist in the two copy databases before and after the division and the latter are connected after the division by a directed relationship of the same type reltype and with the same name rname, but with source data element and target data element reversed. During the reintegration, at least one of these relationships must be rejected or the two must be modified. A directed relationship means a relationship which is directed from a target data element to a source data element.

An anomaly is defined as follows:

**Anomaly:**

An anomaly AM between two conflicts K1(EHA, EHB, GH) and K1(EHA, EHB, GH) is a 4-tuple

AM(K1, K2) *def* (id, amtype, cid1, cid2)

- id is a one-to-one identifier throughout the system (see also definition of a data element)

- amtype $\in$ {Anomaly1a, ..., Anomaly16AB}

SUBSTITUTE SPECIFICATION

- cid1 = K1.id
- cid2 = K2.id.

An anomaly register AMR for two histories is defined as follows:

5 **Anomaly register AMR:**

An anomaly register AMR for two histories EHA and EHB and a common history GH is a 1-tuple

AMR (EHA, EHB, GH) _def_ (anomalyids)

- anomalyids are the identifiers for all anomalies between the histories EHA and
10 EHB and the common history GH.

Pseudo-anomalies describe situations in which the occurrence of an anomaly from conflicts which are present can be prevented only through targeted minimization of the decision options for the conflicts.

15 A pseudo-anomaly is defined as illustrated below:

**Pseudo-anomaly PAM:**

A pseudo-anomaly PAM between two conflicts K1(EHA, EHB, GH) and K2(EHA, EHB, GH) is a 4-tuple

PAM(K1, K2) _def_ (id, pamtype, cid1, cid2)

20 - id is a one-to-one identifier throughout the system (see also definition of a data element)
- pamtype $\in$ {pseudo-Anomaly1A, ..., pseudo-Anomaly32AB}
- cid1 = K1.id
- cid2 = K2.id.

25

A pseudo-anomaly register PAMR is defined as follows:

**Pseudo-anomaly register PAMR:**

A pseudo-anomaly register PAMR for two histories EHA and EHB and a common history GH is a 1-tuple

30 PAMR(EHA, EHB, GH) _def_ (pseudo-anomalyids)

SUBSTITUTE SPECIFICATION

- Pseudo-anomalyids are the identifiers for all pseudo-anomalies between the histories EHA and EHB and the common history GH.

After the conflicts have been ascertained, each ascertained conflict is resolved via a respective individual decision. The conflict resolution process thus comprises a sequence of conflict resolution decisions. The conflict resolution is denoted in Figure 1 by step 104.

In principle, there are various decision options:

a) Adopt the conflict operation(s);

b) Reject the conflict operation(s);

c) Partly adopt, partly reject the conflict operation(s); and

d) Reject the conflict operation(s), adopt new created operation(s).

The individual conflict types are allocated a decision set ES that contains possible decisions which can be used to eliminate an inconsistency created by an operation of the respective conflict type, to which a respective decision set ES is allocated.

Figure 4 provides a compilation of all of the decision sets which are allocated to a respective conflict. Each row of the table shown in Figure 4 shows a possible decision option E1, E2, E3a, E3b, E4, E5a, E5b, and E6. In each case, an x in a field denotes that the conflict shown in the respective column can be resolved by a decision option which is shown in the respective row.

The following text gives an overview of the possible decision options: A first decision option E1 describes the adoption of a conflict operation or of a plurality of conflict operations. A conflict operation describes all of the data operations associated with a conflict. "Adopt" means that the conflict operations are executed in the copy database in which they have not yet been carried out.

A second decision option E2 describes the rejection of a conflict operation or a plurality of conflict operations.

A third decision option E3 describes the adoption of one or more conflict operation(s) in one copy database and the rejection of the conflict operation(s) in the

SUBSTITUTE SPECIFICATION

other copy database.  For the third decision option E3, a detail decision is provided which defines which of the conflict operations present in the histories of different copy databases of the users A and B are to be adopted and which are to be rejected.  These decision options are denoted as a first part E3a of the third decision option E3 and as a second part E3b of the third decision option E3. The first part E3a of the third decision option E3 describes the adoption of the conflict operation(s) in the copy database of the first user A and the rejection of the conflict operation(s) in the copy database of the second user B. The second part E3b of the third decision option E3 describes the adoption of the conflict operation(s) of the copy database of the second user B and the rejection of the conflict operation(s) of the copy database of the first user A.

As Figure 4 shows, a first decision set ES1, which is allocated to the first harmless conflict HK1, describes the first decision option E1 and the second decision option E2 to eliminate the first harmless conflict HK1.  A second decision set ES2 is allocated to the second harmless conflict HK2 and again contains the first decision option E1 and the second decision option E2 to eliminate the second harmless conflict HK2.

If the previous resolution options through the adoption or rejection of available conflict operations do not correspond to the target ideas of the users with regard to the ultimate reintegrated database, i.e., the users A and B cannot agree on one state described by the conflicts, then there is the option of adopting an intermediate solution or the option of selecting and adopting new operations not contained in the decision set. The two options will be illustrated below.

For a conflict with a number n of identical operations, defining the conflict, from a set of data operations in only one history (conflicts of the type HK7 II and HK8 II), there are generally selection options for intermediate states, as described below.

For the seventh harmless conflict HK7 with n = 1, HK7 I is written below, and HK7 II is written below for the seventh harmless conflict HK7 with n > 1. For the eighth harmless conflict HK1 with n = 1, HK8 I is written below, and HK8 II is written below for the eighth harmless conflict HK8 with n > 1.

In this context, a fourth decision option E4 describes a partial adoption and partial rejection of the conflict operations. For the fourth decision option E4, a more precise statement is provided which defines how many of the conflict operations arising on one side are to be adopted and how many are to be rejected. For a

5  number of n operations (n is the element of the natural numbers), the decision options extend from adoption of one operation and rejection of n-1 operations up to adoption of n-1 operations and rejection of one operation. In this case, an adequate decision option is to define the number k (0 < k < n) of adopted operations. The number of rejected operations is then calculated from n - k. The fourth decision

10  option E4 can thus be specialized as follows: the fourth decision option E4 describes the number of adopted conflict operations k.

In addition, the fifth critical conflict KK5 with n = m = 1 is denoted by KK5 I below and the fifth critical conflict KK5 with n > 1 and m > 1 is denoted by KK5 II below.

15  For a fifth critical conflict KK5 II with a number n of identical operations changePropertyAdd, defining the conflict, in one copy database and a number m of identical operations changePropertyDel, defining the conflict, in another copy database, selection of the intermediate states is more difficult.

Since the operations defining the conflict in one history and the operations

20  involved in the conflict in the other history cancel one another out, the same end result can be obtained via different decision options.

Thus, resetting a changePropertyAdd operation and adopting a changePropertyAdd operation in one copy database in conjunction with resetting a changePropertyDel operation in another copy database can obtain the same result

25  as adopting two changePropertyAdd operations in one copy database and a changePropertyDel operation in the other copy database. So that all decision options between the extremes of rejecting the operations in one copy database and adopting the operations in the other copy database (third decision option E3a/E3b) are provided, but various decision options are at the same time prevented from

30  supplying an identical result, there is the solution option of the fifth decision option E5, as illustrated below.

-29-  SUBSTITUTE SPECIFICATION

The fifth decision option E5 can create the last common state between the copy databases by rejecting all operations and permits all other final states from the combinations of the operations, but without the conflict states present above. The fifth decision option E5 thus describes the partial adoption and partial rejection of the conflict operations in one copy database and the rejection of the conflict operations in another copy database.

For the fifth decision option E5, the decision sub-options are necessary, which firstly define which copy database is affected by the partial adoption and partial rejection and which database is affected by the total rejection, and secondly how many of the conflict operations are to be adopted in the case of partial adoption and how many are to be rejected. In this case, the number of operations during partial adoption and partial rejection can be defined as in the case of the fourth decision option E4 using the sole definition of the number of adopted operations. The number of adopted operations in a first copy database of the first user A is in this case denoted by i, and the number of adopted operations in the copy database of the second user B is denoted by k. Thus, detail decisions for the fifth decision option E5 are:

First part E5a of the fifth decision option E5:

> Number of adopted conflict operations for the copy database of the first user A:
>
> $(1 < i < n)$ if a changePropAdd operation is involved,
>
> $(1 < k < m)$ if a changePropDel operation is involved, and rejection of all conflict operations for the copy database of the second user B.

Second part E5b of the fifth decision option E5:

> Number of adopted conflict operations for the copy database of the second user B:
>
> $(1 < i < n)$ if a changePropAdd operation is involved,
>
> $(1 < k < m)$ if a changePropDel operation is involved, and rejection of all conflict operations for the copy database of the first user A.

A selection option for creating a new state for the conflict will be defined by the decision options below. In general, the option of creating and selecting a state

SUBSTITUTE SPECIFICATION

which differs from the two available versions is provided for all conflicts apart from for the conflicts of type HK1, HK2, HK4, HK10.

For creating a new state, it is necessary to create a common starting position, i.e., the relevant operation(s) must be rejected and the two copy databases must be made consistent in terms of the data structure affected by the conflict. This rejection of the operations is not necessary for operations of the changeProperty() type (HK6, HK4) which overwrite one another, because the operation created with the new state overwrites the old operations directly.

For a conflict with an operation defining the conflict from the set of data operations (conflicts of the types HK3, HK6, HK7 I, HK8 I and HK9), for a conflict with a plurality of operations defining the conflict from the set of data operations in the case of just one copy database (conflicts of the types HK5, HK7 II, HK8 II, HK11) and for a conflict with at least one operation defining the conflict from the set of data operations in the case of the two copy databases (conflicts of the types KK1 through KK8), there is the following resolution option, which is called the sixth decision option E6.

The sixth decision option E6 describes the rejection of the conflict operation(s) and selection of new operation(s). In this context, to change a relationship (KK2) on both sides or to change an entry in an ordered listing (KK7) on both sides, the rejection, in contrast to the second decision option E2 (rejection of all operations), relates only to the creator operations for the new relationships or for the new entries. The common deletion operation for the old relationship or for the old entry remains unaffected. For the third critical conflict KK3 and the eighth critical conflict KK8, the sixth decision option E6 has the option only of changing the createRelationship or the changePropertyInsert operation. The common deletion operation remains unaffected.

For the definition of a new state for a conflict of type HK7, HK8 or KK5, the number of

changePropertyAdd operations and

changePropertyDel operations is likewise described with i and k

(i if changePropAdd operations are involved and

k if changePropDel operations are involved).

For the selection of a new state and the creation of the operations necessary for this, interactions on the surface of the application program are customary. If the option exists of selecting a new state using only a complex interaction and if the operations created by the interaction also affect other, unresolved conflicts, then there is the option

a) of using the decision set to resolve the other affected conflicts as well, or

b) of shifting creation of the new state to a later instant, i.e., after reintegration and during the coupled further work.

If, in accordance with a), other affected conflicts are likewise to be resolved, then it is first necessary to reject the operations of the conflict set if conflicts of the type HK6 and KK4 are involved and the operations are not operations which overwrite one another.

Figure 4 shows the decision sets ES1, ES2, ES3, ES4, ES5, ES6, ES7, ES8, ES9, ES10, ES11, ES12, ES13, ES14, ES15, ES16, ES17, ES18, ES19, ES20, ES21, ES22, which are allocated to the respective conflicts.

The sixth harmless conflict HK6 is allocated a sixth decision set ES6, containing the first decision option E1, the second decision option E2 and the sixth decision option E6. A twelfth decision set ES12 is allocated to the first critical conflict KK1. The twelfth decision set ES12 contains four possible decisions, the second decision option E2, the first part E3a of the third decision option E3, the second part E3b of the third decision option E3 and the sixth decision option E6.

The further decision sets are shown in Figure 4 and will be illustrated below for purposes of simplification with the aid of the following list:

- The first harmless conflict HK1, the second harmless conflict HK2, the fourth harmless conflict HK4 and the tenth harmless conflict HK10 are allocated respective decision sets comprising the first decision option E1 and the second decision option E2.

SUBSTITUTE SPECIFICATION

- The third harmless conflict HK3, the fifth harmless conflict HK5, the sixth harmless conflict HK6, the first type HK7 I of the seventh harmless conflict HK7, the first type HK8 I of the eighth harmless conflict HK8, the ninth harmless conflict HK9 and the eleventh harmless conflict HK11 are allocated respective decision sets containing the first decision option E1, the second decision option E2 and the sixth decision option E6.

- The second type HK7 II of the seventh harmless conflict HK7, the second type HK8 II of the eighth harmless conflict HK8, the first critical conflict KK1, the second critical conflict KK2, the third critical conflict KK3, the fourth critical conflict KK4, the first type KK5 I of the fifth critical conflict KK5, the sixth critical conflict KK6, the seventh critical conflict KK7 and the critical conflict KK8 are allocated respective decision sets containing the second decision option E2, the first part E3a of the third decision option E3, the second part E3b of the third decision option E3 and the sixth decision option E6.

- The second type KK5 II of the fifth critical conflict KK5 is allocated a decision set having six possible decisions, the second decision option E2, the first part E3a of the third decision option E3, the second part E3b of the third decision option E3, the first part E5a of the fifth decision option E5, the second part E5b of the fifth decision option E5 and the sixth decision option E6.

- The second type HK7 II of the seventh harmless conflict HK7 and the second type HK8 II of the eighth harmless conflict HK8 are each allocated a decision set having four possible decisions, the first decision option E1, the second decision option E2, the fourth decision option E4 and the sixth decision option E6.

**Restrictions on decision options**

The correct execution of individual decisions is dependent on the presence of a data element or of a plurality of data elements in the two copy databases. By way of example, for adoption in the case of the third harmless conflict HK3A, the two data elements denoted by their identifiers in the relationship operation in conflict must also be present in the copy database of the user B. If one of the data elements is missing, or indeed if both are missing, then this decision about adoption of the

-33-        SUBSTITUTE SPECIFICATION

operation is not possible. It is thus evident that dependencies may exist between individual conflicts.

A dependency of a conflict on conflicts of the type HK1A, HK1B, HK2A or HK2B in terms of its decision options is defined as follows:

5  **Dependent conflict:**

A conflict is dependent on the first harmless conflict HK1 or on the second harmless conflict HK2 if its decision options are restricted by the presence of a first harmless conflict HK1 or of a second harmless conflict HK2. A dependency AK of a conflict is defined as follows:

10 **Dependency AK of a conflict:**

A dependency AK of a conflict K1(EHA, EHB, GH) on a conflict K1(EHA, EHB, GH) is a 4-tuple

AK (K1, K2) *def* (id, ctype, cid1, cid2)

- id is a one-to-one identifier throughout the system (see also definition of a data
15  element)
- ctype ∈ {HK1A, HK1B, HK2A, HK2B}
- cid1 = K1.id
- cid2 = K2.id.

20 A dependency register AKR is defined as follows:

**Dependency register AKR:**

A dependency register AKR for two histories EHA and EHB and a common history GH is a 1-tuple

AKR (EH$_A$, EH$_B$, GH) *def* (dependencyids)

25 - dependencyids are the identifiers for all recognized dependencies of the histories EH$_A$ and EH$_B$ and the common history GH.

All restrictions on decision options by existing conflicts of the first harmless conflict HK1 and of the second harmless conflict HK2 are recognized and marked at the start of conflict resolution on the basis of the dependencies AK. This is done

30 using the decision restrictions' parameter introduced in the conflict definition for each conflict. All of the possible restrictions are described below.

–34–  SUBSTITUTE SPECIFICATION

The first harmless conflict HK1 impairs the decision options for conflicts with operations within the dedicated history. These dependent conflicts include all those containing a createRelationship operation or a property change with the created data element. In this context, in the case of harmless conflicts HK6 through HK11 with property operations for this data element, the decisions are minimized by an adoption and a creation of a new state.

The decision options for the critical conflicts KK1, KK2 and KK3 with relationship operations with the created data element are reduced by the option of adopting the operations (-E3a, -E3b) for the copy database whose history contains the createElement operation. The decision options for the third harmless conflict HK3 and the fifth harmless conflict HK5 with relationship operations with the created data element are reduced by the decision to accept and/or create a new state. Critical conflicts with property operations experience no modifications to their decision collection (decision sets).

A second harmless conflict HK2 impairs the decision option for conflicts in the two copy databases. The harmless conflicts for modifying properties HK6 through HK11 are treated as for the first harmless conflict HK1. The harmless conflicts with relationship operations (HK4, HK5) in the history of the deleteElement operation no longer have any decision option for rejection and the decisions relating to the harmless relationship conflicts in the other copy database HK3, HK5 are minimized by the option of accepting and/or creating a new state. The critical conflicts with relationship operations in the copy database without the deleteElement operation are reduced by the option resetting and/or adopting the operations. Critical conflicts with property operations experience no modification to their property collection in this case either.

A common deleteElement operation contained in the common history GH and representing no conflict reduces the decision options in specific cases. Thus, all critical conflicts KK2 and KK3 with relationship operations in which the target data element of the common deleteRelationship operation corresponds to the data element deleted on both sides can no longer be reset.

Dependent conflicts with one or more relationship operations (HK3, HK4, HK5, KK1, KK2, KK3), i.e., with a plurality of listed identifiers and hence a plurality of data elements involved, can have a plurality of dependencies at the same time. In this regard, there may be just one dependency for each arising identifier in a conflict.

5 A dependent conflict having a plurality of identifiers can firstly have a plurality of dependencies on conflicts of the same conflict type (for example, on two conflicts of the first harmless conflict HK1a) and can secondly have a plurality of dependencies on conflicts of different conflict types (e.g., on a conflict of the type HK1a and on a conflict of the type HK1b). By way of example, a dependent conflict of the type HK3a

10 can have two dependencies on two conflicts of the type HK1a, namely one with the identifier id1 and one with the identifier id2. At the same time, the second critical contact KK2 can have a dependency on the second harmless conflict HK2a, on the first harmless conflict HK1a and on the first harmless conflict HK1B.

The dependency of the identifiers for a conflict of the type HK1 or HK2 means

15 that there can be a maximum of three restrictions per conflict of the type HK5, KK1, KK2 or KK3. All conflicts of the type HK3A, HK3B, HK4A, HK4B, HK5A and HK5B can have a plurality of dependencies on conflicts of the same or different type at the same time. Conflicts of the types KK1, KK2, KK3A and KK3B, on the other hand, can have a plurality of dependencies on conflicts of different types at the same time.

20 However, the same restrictions of the decision options may also arise a number of times for conflicts which have a number of dependencies on the same conflict type. Thus, for a conflict of the type HK3A, two dependencies are possible at the same time in the presence of a conflict of the type HK1A or of one of the type HK2B and the decisions are each minimized by the first decision E1.

25 The multiple restrictions arising for many dependent conflicts with the same decision options do not require any separate consideration. Each of these decision restrictions is considered as if it were an individual, specific limitation. Hence, all multiple restrictions, like others as well, are noted in the conflict. If a conflict of the type HK1 and HK2 is resolved such that one of the restrictions arising a number of

30 times is eliminated, the rest of these restrictions are retained. Only when different conflict resolutions mean that there are no longer any of the multiple restrictions

SUBSTITUTE SPECIFICATION

present can a decision of this type be made. This applies irrespective of whether the restriction was present once or a plurality of times.

The restrictions, recognized once at the start of reintegration, as a result of dependencies of the conflicts on conflicts of the type HK1 and HK2 are changed dynamically depending on the resolution of the conflicts of the type HK1 and HK2 during the reintegration. Thus, depending on the type of the dependent conflicts and on the respective resolution decision for the createElement operation and deleteElement operation, the following changes to the decision restrictions result:

a) The adoption of a createElement operation (first decision E1 relating to a conflict of the type HK1) causes the decision restrictions to be reset, i.e., the decision options to be extended, for all conflicts dependent on the operation.

b) The rejection of a deleteElement operation (second decision E2 relating to a conflict of the type HK2) likewise causes the decision restrictions to be reset, i.e., the decision options to be extended for the conflicts dependent on this conflict.

c) The rejection of a createElement operation (second decision E2 relating to a conflict of the type HK1) causes the decision restrictions already made for the conflicts dependent on this conflict to be retained. Hence, no decision options change.

d) The adoption of a deleteElement operation (first decision E1 relating to a conflict of the type HK2) causes the decision restrictions already made for the conflicts dependent on this conflict to be retained. Hence, no decision options change.

An anomaly created on one side can be recognized by way of two respective conflicts present in a history. In this context, there are the options of the conflict pairs HK5Aa/HK5Ab, HK4A/HK3A, HK4A/HK5A. In the case of a conflict of the type HK5Aa, the deleteRelationship operation is involved in the anomaly and in the case of a conflict of the type HK5Ab, the createRelationship operation is involved in the anomaly.

For anomalies arising on account of modifications on the copy database of the second user B, the same options apply: HK5Ba/HK5Bb, HK4B/HK3B,

HK4B/HK5B. In this context, a common feature of all conflict pairs is that there is a deleteRelationship operation with an identifier id1 as a source data element and with an identifier id2 as a target data element in one of the two conflicts and the same identifiers arise in reverse as source data element and target data element in a createRelationship operation for the other conflict.

The adoption of the conflict of the one-sided anomaly, as described above, with the createRelationship operation prevents rejection of the conflict of the one-sided anomaly with the deleteRelationship operation for a rejection of the conflict of the one-sided anomaly with the deleteRelationship operation, but on the other hand prevents adoption of the conflict of the anomaly with the createRelationship operation. The change in one of the two conflicts does not reduce the decision options for the other conflict.

The result of this is that an anomaly created on one side in directed relationships can be resolved by rejecting the two conflicts, adopting the two conflicts, changing the two conflicts differently or modifying one of the conflicts and adopting or rejecting the other conflict.

An anomaly created on both sides in directed relationships can be resolved by deciding one of the two conflicts and deciding the other conflict in a different manner from this.

To prevent an anomaly (a "pseudo-anomaly") from arising, as described above, the following restrictions on the decision options are provided:

a) After adoption of a conflict with the createRelationship operation containing the common data element id1 as a target data element (createRelationship21), there must no longer be, for the conflict with the two createRelationship operations containing the common data element as source data element, any decision option of the sixth decision option E6 with replacement of the target data element (idx or idz) by the source data element id2(createRelationship21).

b) After a sixth decision E6 has been made on the basis of the sixth decision option E6 and selection of a new target data element id2 for the conflict with the two createRelationship operations containing the common data element id1 as a source data element, no further adoption of the conflict with the

createRelationship operation with the common data element id1 as a target data element (createRelationship21) must be possible.

As described above, as part of this method, the decision options for conflicts are limited on the basis of their dependencies, anomalies and pseudo-anomalies. After each decision relating to a conflict, a change to the decision options for the conflicts which are dependent on the conflict just resolved or are situated in a common anomaly or pseudo-anomaly with this conflict is made on the basis of the dependencies, anomalies and pseudo-anomalies.

A decision is made for each conflict, as described above. The decision can be made in different ways. An overview of possible decision variations can be found in '132. Within the scope of this illustrative embodiment, provision is made for a database or copy database to be regarded as a reference database, and for the adjustment to be carried out on the basis of the reference database.

Thus, as shown in Figure 1 by way of a recursive loop via a checking step (step 105), a check is made to determine whether there is still any conflict and whether a decision thus needs to be made, and to make this decision if needed. If there are no more conflicts present, then a last method step (step 106) is carried out, storage of the reintegrated database, which contains no more inconsistencies.

The inconsistency-free database is again transmitted to all further computers connected to the first computer 200 (step 107). All the computers therefore have a consistent copy database.

The following text illustrates a few alternatives to the illustrative embodiment described above. Inconsistencies can also be recognized after a prescribable number of elimination operations carried out for an inconsistency by searching for a further inconsistency. This can be extended such that, each time an inconsistency has been eliminated, a subsequent inconsistency is again sought and eliminated. It is also possible for the first computer to use the method illustrated above to ascertain a series of correction commands (correction sequences) which is respectively transmitted to the computer whose copy database has been checked

for inconsistencies, and for the respective computer to use the correction sequence to adjust its copy database to the database.

In addition, in one alternative embodiment, it is likewise possible to leave the decision to a user or to a plurality of users, i.e., the decision options are displayed to a user on the screen and the user uses the keyboard or the computer mouse to select the decision he requires, which is then implemented by the computer.

The above-described method and arrangement are illustrative of the principles of the present invention. Numerous modifications and adaptations thereof will be readily apparent to those skilled in this art without departing from the spirit and scope of the present invention.

SUBSTITUTE SPECIFICATION

ABSTRACT

To eliminate at least one inconsistency in a database collection containing a
database and at least one copy database of the database, which inconsistency
arises on account of the data in the database or in the copy database being
5   changed, at least some of the operations which can create an inconsistency are
allocated to defined conflict types. Each conflict type is allocated a decision set
which is used to indicate possible decisions which can be used to eliminate an
inconsistency created by an operation of the respective conflict type. The
inconsistency is eliminated using the decision set. Error-free elimination of
10  inconsistencies is ensured by matching the decision set to the respective situation.

J002 Rec'd PCT/PTO   2 2 JAN 2001

BOX PCT
IN THE UNITED STATES DESIGNATED/ELECTED OFFICE
OF THE UNITED STATES PATENT AND TRADEMARK OFFICE
UNDER THE PATENT COOPERATION TREATY–CHAPTER II

CHANGE OF ADDRESS OF APPLICANTS' REPRESENTATIVE

APPLICANT(S):                           MICHAEL BERGER

ATTORNEY DOCKET NO.:                    P00,1950

INTERNATIONAL APPLICATION NO:           PCT/DE99/01999

INTERNATIONAL FILING DATE:              01 JULY 1999

INVENTION:     METHOD, ARRANGEMENT AND SET OF A PLURALITY OF
               ARRANGEMENTS FOR ELIMINATING AT LEAST ONE INCONSISTENCY
               IN A DATABASE COLLECTIION CONTAINING A DATABASE AND AT
               LEAT ONE COPY DATABASE OF THE DATABASE

Assistant Commissioner for Patents,
Washington D.C. 20231

S I R:

       Members of the firm of Hill & Simpson designated on the original Power of Attorney have

merged into the firm of Schiff Hardin & Waite. All future correspondence for the above-referenced

application therefore should be sent to the following address:

                         SCHIFF HARDIN & WAITE
                           Patent Department
                            6600 Sears Tower
                         233 South Wacker Drive
                      Chicago, Illinois 60606-6473


              Submitted by,


              _____          (Reg. No. 45,877)
              Mark Bergner
              SCHIFF HARDIN & WAITE
              Patent Department
              6600 Sears Tower
              Chicago, Illinois 60606-6473
              Telephone: (312) 258-5779
              Attorneys for Applicants

09/744225

BOX PCT
IN THE UNITED STATES DESIGNATED/ELECTED OFFICE
OF THE UNITED STATES PATENT AND TRADEMARK OFFICE
UNDER THE PATENT COOPERATION TREATY--CHAPTER II

APPLICANT(S):                          MICHAEL BERGER

ATTORNEY DOCKET NO.:              P00,1950

INTERNATIONAL APPLICATION NO:   PCT/DE99/01999

INTERNATIONAL FILING DATE:        01 JULY 1999
INVENTION:        METHOD, ARRANGEMENT AND SET OF A PLURALITY
                  OF ARRANGEMENTS FOR ELIMINATING AT LEAST ONE
                  INCONSISTENCY IN A DATABASE COLLECTION
                  CONTAINING A DATABASE AND AT LEAST ONE COPY
                  DATABASE OF THE DATABASE

Assistant Commissioner for Patents,
Washington, D.C. 20231

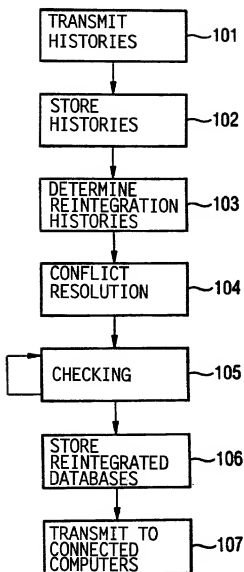## REQUEST FOR APPROVAL OF DRAWING ADDITIONS

Sir:

Enclosed are 4 sheets of drawings, Figures 1-4, showing in red,
the addition of labels to the elements depicted in Figures 1, 2 & 3.
Approval of the additions is respectfully requested.

Submitted by,

_____   (Reg. No. 45,877)
Mark Bergner
SCHIFF HARDIN & WAITE
PATENT DEPARTMENT
6600 Sears Tower
Chicago, Illinois 60606-6473
(312) 258-5779
Attorney for Applicant(s)

**FIG 1**

```
┌─────────────┐
│  TRANSMIT   │──101
│  HISTORIES  │
└─────────────┘
       │
       ▼
┌─────────────┐
│   STORE     │──102
│  HISTORIES  │
└─────────────┘
       │
       ▼
┌──────────────┐
│  DETERMINE   │──103
│ REINTEGRATION│
│  HISTORIES   │
└──────────────┘
       │
       ▼
┌─────────────┐
│  CONFLICT   │──104
│ RESOLUTION  │
└─────────────┘
       │
       ▼
┌─────────────┐
│  CHECKING   │──105
└─────────────┘
       │
       ▼
┌──────────────┐
│   STORE      │──106
│ REINTEGRATED │
│  DATABASES   │
└──────────────┘
       │
       ▼
┌──────────────┐
│ TRANSMIT TO  │──107
│  CONNECTED   │
│  COMPUTERS   │
└──────────────┘
```

**FIG 2**

FIG 3

DATABASE
301
302
303
304
305
306
307
308
309

FIRST COPY DATABASE
310
311
312
313
314
315
316
317
318

SECOND COPY DATABASE
320
321
322
323
325
326
327
328

**FIG 4**

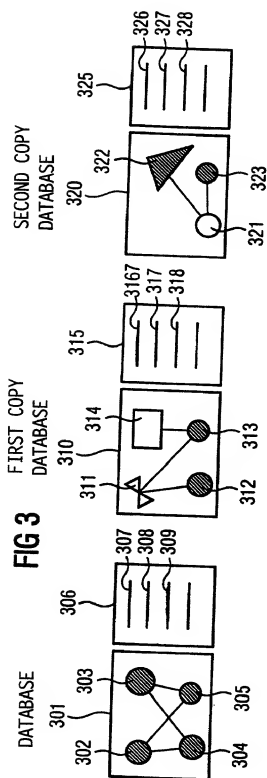| | ES1 H K1 | ES2 H K2 | ES3 H K3 | ES4 H K4 | ES5 H K5 | ES6 H K6 | ES7 H K 7I | ES8 H K 7II | ES9 H K 8I | ES10 H K 8II | ES11 H K9 | ES12 H K 10 | ES13 H K 11 | ES14 K K1 | ES15 K K2 | ES16 K K3 | ES17 K K4 | ES18 K K 5I | ES19 K K 5II | ES20 K K6 | ES21 K K7 | ES22 K K8 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| E1 | × | × | × | × | × | × | × | × | × | × | × | × | × | | | | | | | | | |
| E2 | × | × | × | × | × | × | × | × | × | × | × | × | × | × | × | × | × | × | × | × | × | × |
| E3a | | | | | | | | | | | | | | × | × | × | × | × | × | × | × | × |
| E3b | | | | | | | | | | | | | | × | × | × | × | × | × | × | × | × |
| E4 | | | | | | | | (×) | | × | | | | | | | | | | | | |
| E5a | | | | | | | | | | | | | | | | | | | × | | | |
| E5b | | | | | | | | | | | | | | | | | | | × | | | |
| E6 | | | × | | × | × | × | × | × | × | × | | × | × | × | × | × | × | × | × | × | × |

# Declaration and Power of Attorney For Patent Application
## *Erklärung Für Patentanmeldungen Mit Vollmacht*
### German Language Declaration

Als nachstehend benannter Erfinder erkläre ich hiermit an Eides Statt:

As a below named inventor, I hereby declare that:

dass mein Wohnsitz, meine Postanschrift, und meine Staatsangehörigkeit den im Nachstehenden nach meinem Namen aufgeführten Angaben entsprechen,

My residence, post office address and citizenship are as stated below next to my name,

dass ich, nach bestem Wissen der ursprüngliche, erste und alleinige Erfinder (falls nachstehend nur ein Name angegeben ist) oder ein ursprünglicher, erster und Miterfinder (falls nachstehend mehrere Namen aufgeführt sind) des Gegenstandes bin, für den dieser Antrag gestellt wird und für den ein Patent beantragt wird für die Erfindung mit dem Titel:

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled

<u>Verfahren, Anordnung und Satz mehrerer Anordnungen zur Behebung mindestens einer Konsistenz in einer Datenbankmenge, die eine Datenbank sowie mindestens eine Kopiedatenbank der Datenbank aufweist</u>

_____

_____

_____

_____

deren Beschreibung

the specification of which

(zutreffendes ankreuzen)

(check one)

☒ hier beigefügt ist.

☐ is attached hereto.

☐ am _____ als
PCT internationale Anmeldung
PCT Anmeldungsnummer _____
eingereicht wurde und am
abgeändert wurde (falls tatsächlich abgeändert).

☐ was filed on _____ as
PCT international application
PCT Application No. _____
and was amended on _____
(if applicable)

Ich bestätige hiermit, dass ich den Inhalt der obigen Patentanmeldung einschließlich der Ansprüche durchgesehen und verstanden habe, die eventuell durch einen Zusatzantrag wie oben erwähnt abgeändert wurde.

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims as amended by any amendment referred to above.

Ich erkenne meine Pflicht zur Offenbarung irgendwelcher Informationen, die für die Prüfung der vorliegenden Anmeldung in Einklang mit Absatz 37, Bundesgesetzbuch, Paragraph 1.56(a) von Wichtigkeit sind, an.

I acknowledge the duty to disclose information which is material to the examination of this application in accordance with Title 37, Code of Federal Regulations, §1.56(a).

Ich beanspruche hiermit ausländische Prioritätsvorteile gemäss Abschnitt 35 der Zivilprozessordnung der Vereinigten Staaten, Paragraph 119 aller unten angegebenen Auslandsanmeldungen für ein Patent oder eine Erfindersurkunde, und habe auch alle Auslandsanmeldungen für ein Patent oder eine Erfindersurkunde nachstehend gekennzeichnet, die ein Anmeldedatum haben, das vor dem Anmeldedatum der Anmeldung liegt, für die Priorität beansprucht wird.

I hereby claim foreign priority benefits under Title 35, United States Code, §119 of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

Form PTO-FB-240 (8-83)                    Patent and Trademark Office-U.S. DEPARTMENT OF COMMERCE

# German Language Declaration

| 198 33 778.7 | Germany | 27. Juli 1998 | [X] | [ ] |
|---|---|---|---|---|
| (Number) | (Country) | (Day Month Year Filed) | Yes | No |
| (Nummer) | (Land) | (Tag Monat Jahr eingereicht) | Ja | Nein |

| | | | [ ] | [ ] |
|---|---|---|---|---|
| (Number) | (Country) | (Day Month Year Filed) | Yes | No |
| (Nummer) | (Land) | (Tag Monat Jahr eingereicht) | Ja | Nein |

| | | | [ ] | [ ] |
|---|---|---|---|---|
| (Number) | (Country) | (Day Month Year Filed) | Yes | No |
| (Nummer) | (Land) | (Tag Monat Jahr eingereicht) | Ja | Nein |

Ich beanspruche hiermit gemäss Absatz 35 der Zivilprozessordnung der Vereinigten Staaten, Paragraph 120, den Vorzug aller unten aufgeführten Anmeldungen und falls der Gegenstand aus jedem Anspruch dieser Anmeldung nicht in einer früheren amerikanischen Patentanmeldung laut dem ersten Paragraphen des Absatzes 35 der Zivilprozeßordnung der Vereinigten Staaten, Paragraph 122 offenbart ist, erkenne ich gemäss Absatz 37, Bundesgesetzbuch, Paragraph 1.56(a) meine Pflicht zur Offenbarung von Informationen an, die zwischen dem Anmeldedatum der früheren Anmeldung und dem nationalen oder PCT internationalen Anmeldedatum dieser Anmeldung bekannt geworden sind.

I hereby claim the benefit under Title 35. United States Code. §120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, §122, I acknowledge the duty to disclose material information as defined in Title 37, Code of Federal Regulations, §1.56(a) which occured between the filing date of the prior application and the national or PCT international filing date of this application.

| (Application Serial No.) | (Filing Date) | (Status) | (Status) |
|---|---|---|---|
| (Anmeldeseriennummer) | (Anmeldedatum) | (patentiert, anhängig, aufgegeben) | (patented, pending, abandoned) |

| (Application Serial No.) | (Filing Date) | (Status) | (Status) |
|---|---|---|---|
| (Anmeldeseriennummer) | (Anmeldedatum) | (patentiert, anhängig, aufgeben) | (patented, pending, abandoned) |

Ich erkläre hiermit, dass alle von mir in der vorliegenden Erklärung gemachten Angaben nach meinem besten Wissen und Gewissen der vollen Wahrheit entsprechen, und dass ich diese eidesstattliche Erklärung in Kenntnis dessen abgebe, dass wissentlich und vorsätzlich falsche Angaben gemäss Paragraph 1001, Absatz 18 der Zivilprozessordnung der Vereinigten Staaten von Amerika mit Geldstrafe belegt und/oder Gefängnis bestraft werden koennen, und dass derartig wissentlich und vorsätzlich falsche Angaben die Gültigkeit der vorliegenden Patentanmeldung oder eines darauf erteilten Patentes gefährden können.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true, and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

# German Language Declaration

And I hereby appoint

Messrs. John D. Simpson (Registration No. 19,842) Lewis T. Steadman (17,074), William C. Stueber (16,453), P. Phillips Connor (19,259), Dennis A. Gross (24,410), Marvin Moody (16,548), Steven H. Noll (28,982), Brett A. Valiquet (27,841), Thomas I. Ross (29,275), Kevin W. Guynn (29,927), Edward A. Lehmann (22,312), James D. Hobart (24,149), Robert M. Barrett (30,142), James Van Santen (16,584), J. Arthur Gross (13,615), Richard J. Schwarz (13,472) and Melvin A. Robinson (31,870), David R. Metzger (32,919), John R. Garrett (27,888) all members of the firm of Hill, Steadman & Simpson, A Professional Corporation.

| Telefongespräche bitte richten an: *(Name und Telefonnummer)* | Direct Telephone Calls to: *(name and telephone number)* |
|---|---|
| | 312/876-0200 Ext. _____ |

| Postanschrift: | Send Correspondence to: |
|---|---|

**HILL, STEADMAN & SIMPSON**
**A Professional Corporation**
**85th Floor Sears Tower, Chicago, Illinois 60606**

| Voller Name des einzigen oder ursprünglichen Erfinders: | Full name of sole or first inventor: |
|---|---|
| BERGER, Michael | |
| Unterschrift des Erfinders *Michael Berger* Datum 25.5.99 | Inventor's signature Date |
| Wohnsitz D-85635 Höhenkirchen, Germany *DEX* | Residence |
| Staatsangehörigkeit Bundesrepublik Deutschland | Citizenship |
| Postanschrift Am Grenzweg 2 D-85635 Höhenkirchen Bundesrepublik Deutschland | Post Office Addess |
| Voller Name des zweiten Miterfinders (falls zutreffend): | Full name of second joint inventor, if any: |
| Unterschrift des Erfinders Datum | Second Inventor's signature Date |
| Wohnsitz | Residence |
| Staatsangehörigkeit | Citizenship |
| Postanschrift | Post Office Address |
| | |

| *(Bitte entsprechende Informationen und Unterschriften im Falle von dritten und weiteren Miterfindern angeben).* | *(Supply similar information and signature for third and subsequent joint inventors).* |
|---|---|

Form PTO-FB-240 (8-83)                                   Patent and Trademark Office-U.S. Department of COMMERCE